Tran-SET

Transportation Consortium of South-Central States

Solving Emerging Transportation Resiliency, Sustainability, and Economic Challenges through the Use of Innovative Materials and Construction Methods: From Research to Implementation

Decision-Making Tool for Road Preventive Maintenance Using Vehicle Vibration Data

Project No. 18PLSU08

Lead University: Texas A&M University

Collaborative Universities: Louisiana State University

Final Report

August 2019

Disclaimer

The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the information presented herein. This document is disseminated in the interest of information exchange. The report is funded, partially or entirely, by a grant from the U.S. Department of Transportation's University Transportation Centers Program. However, the U.S. Government assumes no liability for the contents or use thereof.

Acknowledgements

The PIs would like to acknowledge the contribution of the graduate student assistant, Zishuo Li, who has been working on this project since it started. Also, we are also grateful for the constructive comments provided by Dr. Jim Ji, the associate professor of Electrical and Computer Engineering at Texas A&M University.

TECHNICAL DOCUMENTATION PAGE

1. Project No.	2. Government Accession No.	3. Recipient's Catalog No.
18PLSU08		

4. Title and Subtitle	5. Report Date
Decision-Making Tool for Road Preventive Maintenance Using Vehicle Vibration Data	Aug. 2019
	6. Performing Organization Code
7. Author(s)	**8. Performing Organization Report No.**
PI: Changbum Ahn https://orcid.org/0000-0002-6733-2216 Co-PI: Chao Wang https://orcid.org/0000-0002-9322-7778 Co-PI: Jing Du (previous PI) https://orcid.org/0000-0002-0481-4875	
9. Performing Organization Name and Address	**10. Work Unit No. (TRAIS)**
Transportation Consortium of South-Central States (Tran-SET) University Transportation Center for Region 6 3319 Patrick F. Taylor Hall, Louisiana State University, Baton Rouge, LA 70803	**11. Contract or Grant No.** 69A3551747106
12. Sponsoring Agency Name and Address	**13. Type of Report and Period Covered**
United States of America Department of Transportation Research and Innovative Technology Administration	Final Research Report Mar. 2018 – Mar. 2019
	14. Sponsoring Agency Code

15. Supplementary Notes

Report uploaded and accessible at Tran-SET's website (http://transet.lsu.edu/).

16. Abstract

Automated and timely road pavement damage inspection is critical to the preventive maintenance and the long-term sustainability and resilience of roads in Region 6. Current road inspection practices rely heavily on a manual process. Sensor-based methods (e.g., LiDAR scanning) are promising but can be too expensive for a wider adoption. This study employs a crowdsourcing approach of using the vibration patterns of regular vehicles in inferring specific types of road damages. A cloud-based smart phone app and system was developed to collect real-time vehicle vibrations, location data, and road damage images for training the detection model. However, there is a great challenge in using classic classification methods with crowdsourced vibration data containing high level of noises, as vehicle vibrations are greatly affected by the types and conditions of the vehicles, as well the varying driving behaviors of drivers. The study thus employed the recent developments in Deep Learning methods, including a Self-Taught Learning (STL) algorithm and Sparse Coding to tackle with the low-quality issues of collected data. A total of 310 miles of road-induced vehicle vibration data was collected in Texas and Louisiana, and the road damage detection model was trained on Texas A&M University (TAMU) supercomputing server. The results show that the features generated from Sparse Coding greatly contribute to enhancing detection performances, by addressing low-quality data issues.

17. Key Words	18. Distribution Statement
Road Inspection, Deep Learning, Vibration	No restrictions. This document is available through the National Technical Information Service, Springfield, VA 22161.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	34	

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized.**

SI* (MODERN METRIC) CONVERSION FACTORS

APPROXIMATE CONVERSIONS TO SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
in	inches	25.4	millimeters	mm
ft	feet	0.305	meters	m
yd	yards	0.914	meters	m
mi	miles	1.61	kilometers	km
		AREA		
in^2	square inches	645.2	square millimeters	mm^2
ft^2	square feet	0.093	square meters	m^2
yd^2	square yard	0.836	square meters	m^2
ac	acres	0.405	hectares	ha
mi^2	square miles	2.59	square kilometers	km^2
		VOLUME		
fl oz	fluid ounces	29.57	milliliters	mL
gal	gallons	3.785	liters	L
ft^3	cubic feet	0.028	cubic meters	m^3
yd^3	cubic yards	0.765	cubic meters	m^3
		NOTE: volumes greater than 1000 L shall be shown in m^3		
		MASS		
oz	ounces	28.35	grams	g
lb	pounds	0.454	kilograms	kg
T	short tons (2000 lb)	0.907	megagrams (or "metric ton")	Mg (or "t")
		TEMPERATURE (exact degrees)		
^oF	Fahrenheit	5 (F-32)/9 or (F-32)/1.8	Celsius	^oC
		ILLUMINATION		
fc	foot-candles	10.76	lux	lx
fl	foot-Lamberts	3.426	$candela/m^2$	cd/m^2
		FORCE and PRESSURE or STRESS		
lbf	poundforce	4.45	newtons	N
lbf/in^2	poundforce per square inch	6.89	kilopascals	kPa

APPROXIMATE CONVERSIONS FROM SI UNITS

Symbol	When You Know	Multiply By	To Find	Symbol
		LENGTH		
mm	millimeters	0.039	inches	in
m	meters	3.28	feet	ft
m	meters	1.09	yards	yd
km	kilometers	0.621	miles	mi
		AREA		
mm^2	square millimeters	0.0016	square inches	in^2
m^2	square meters	10.764	square feet	ft^2
m^2	square meters	1.195	square yards	yd^2
ha	hectares	2.47	acres	ac
km^2	square kilometers	0.386	square miles	mi^2
		VOLUME		
mL	milliliters	0.034	fluid ounces	fl oz
L	liters	0.264	gallons	gal
m^3	cubic meters	35.314	cubic feet	ft^3
m^3	cubic meters	1.307	cubic yards	yd^3
		MASS		
g	grams	0.035	ounces	oz
kg	kilograms	2.202	pounds	lb
Mg (or "t")	megagrams (or "metric ton")	1.103	short tons (2000 lb)	T
		TEMPERATURE (exact degrees)		
^oC	Celsius	1.8C+32	Fahrenheit	^oF
		ILLUMINATION		
lx	lux	0.0929	foot-candles	fc
cd/m^2	$candela/m^2$	0.2919	foot-Lamberts	fl
		FORCE and PRESSURE or STRESS		
N	newtons	0.225	poundforce	lbf
kPa	kilopascals	0.145	poundforce per square inch	lbf/in^2

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS, ABBREVIATIONS, AND SYMBOLS

ASCE	American Society of Civil Engineers
CNN	Convolutional Neural Network
DOT	Department of Transportation
FHA	Federal Highway Administration
FP	False Positives
FN	False Negatives
GPR	Ground Penetration Radar
IRI	International Roughness Index
MLP	Multilayer Perceptron
RVV	Road Vehicle Vibration
SVM	Support Vector Machine
STL	Self-Taught Learning
TAMU	Texas A&M University
TN	True Negatives
TP	True Positives

EXECUTIVE SUMMARY

America's aging infrastructure systems are facing a significant challenge due to the limited renovation funding. Given the expanding capital gap, the American Society of Civil Engineers (ASCE) states that "*fostering optimization of infrastructure investments for society*" is one of the two grand civil engineering challenges in the 21st century. The most cost-effective strategy to improve the overall conditions of America's road infrastructure is through the *Preventive Maintenance*, i.e., a planned treatment to an existing roadway system and its appurtenances before deficiencies develop. Making robust preventive maintenance decisions can be a nontrivial task, due to various factors that need to be considered in the calculation. Even in a small area, different sections of a road can be in varying conditions and under continuous changes. The temporal data of road damages are critical to the prediction of future performance. As a result, effective preventive decision-making requires large amounts of high-quality live data about the road pavement conditions, especially the critical damages that can lead to major maintenance problems.

Road damage data is usually collected during the road inspection. The current practice of road inspection relies heavily on a manual process. Region 6 DOTs have published inspection standards such as Texas Pavement Management Information System Rater's Manual to guide the manual inspection. However, the reliability of this manual inspection process is often affected by the potential evaluation bias of inspectors, the inconsistent evaluation criteria due to the subjective interpretation of the standards, and the limited inspection frequency and coverage due to the shortage of workforce. DOTs have also applied the automated sensing technologies in road damage inspection, such as LiDAR scanning, ground penetration radar, and imagery sensors. These sensing methods, although accurate and efficient, are often too expensive to deploy in a larger area to have a wide impact on the sustainability and resilience of Region 6 roads.

To overcome the foreseeable challenges in system-level road maintenance decision-making, this study aims to test a method, system, and corresponding algorithms that predict major road damages based on running vehicles' vibration data, via sensors built in most smartphones, to collect high resolution live data about road conditions for better preventive maintenance decision-making. The technical objectives include: (1) Develop and test a vehicle vibration-based road damage data collection system; (2) Model the relationship between vehicle vibrations and the ground truth of major road damages; (3) Validate and test the road damage detection model enabled by vehicle vibration patterns.

This study will advance our knowledge about deep learning-enabled automated road damage inspection via a crowdsourcing system. The findings will enable the development of an intelligent and economic road condition evaluation method by monitoring the vibration patterns of running vehicles on the road. Based on that, decision-makers will be able to develop precise road deterioration models to predict the temporal change of conditions. It will ultimately help optimize the preventive maintenance activities to preserve the functional condition of the road system at the minimum cost, without significantly increasing the structural capacity. This has the potential to completely transform our current infrastructure maintenance science from an ad hoc to a more data-driven paradigm. In the same vein, the findings are expected to improve our understanding of the long-term durability of road infrastructure. A large amount of empirical data collected in this study will help build the holistic view of road infrastructure system in the selected areas. The maintenance decision-making will be driven by the system science of all interdependent factors instead of local optimization. Eventually, this study will help promote societal efforts in citizen

science and engineering, i.e., relying on the intelligence of regular population (e.g., car drivers) in challenging engineering problems that machines cannot solve alone.

1. INTRODUCTION

America's aging infrastructure systems are facing a significant challenge due to the limited renovation funding. The estimated cost of fixing America's failing infrastructure is $3.6 trillion by 2020, with only 55% committed *(1)*. Given the expanding capital gap, the American Society of Civil Engineers (ASCE) states that "fostering optimization of infrastructure investments for society" is one of the two grand civil engineering challenges in the 21st century *(2)*. Among all the infrastructure systems, road infrastructure is the one that is most directly related to everybody's daily life. Unfortunately, according to the latest Infrastructure Report Card *(3)*, America's road infrastructure only received a GPA of D. For Region 6, there are total 659,404 miles of public roads with average 22% in poor conditions *(3)*. The Federal Highway Administration estimated that $170 billion in capital investment would be needed on an annual basis to significantly improve the conditions of road infrastructure *(4)*. The deteriorating road condition is also the main contributor to traffic accidents. According to a US DOT report to Congress *(5)*, in the US, 12.2% of auto accidents were attributed to the poor road conditions. In Region 6, the average high temperature in summers can present an additional challenge to the road condition: as illustrated in Figure 1, high temperature can speed up the pavement distress development, causing major road damages that may significantly affect driving comfort and safety *(6)*.

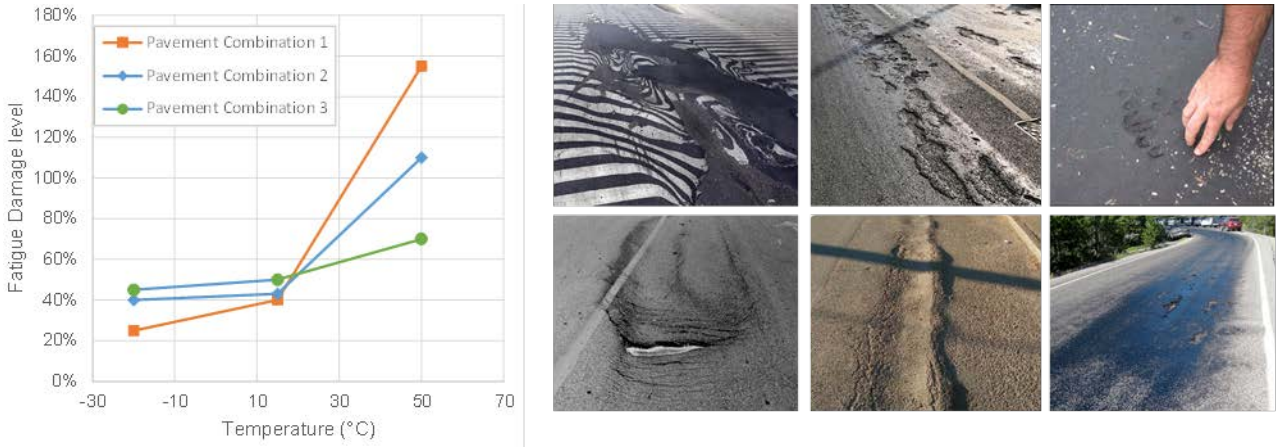

Figure 1. Region 6's high temperature speeds up road damages, left: High temperature speeds up road damage (adapted from (6)), right: Typical road damages in high-temperature regions (source: internet).

The most cost-effective strategy to improve the overall conditions of America's road infrastructure is through the *Preventive Maintenance*, i.e., a planned treatment to an existing roadway system and its appurtenances before deficiencies develop *(7)*. Empirical and laboratory data indicate that the development of pavement distress follows a predictable course *(8)*. Based on the established knowledge of pavement distress development, preventive maintenance aims to preserve the system, retard deterioration, and maintain the functional condition of the system without significantly increasing the structural capacity (Figure 2) *(9)*.

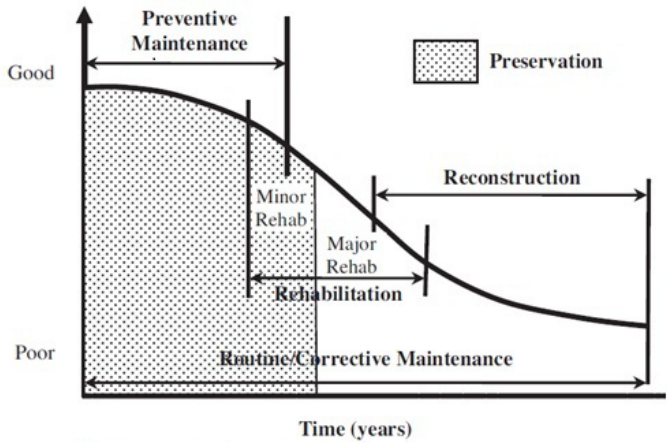

Figure 2. Preventive Maintenance is the most cost-effective way of preserving the existing road infrastructure (Adapted from *(9)*).

However, making robust preventive maintenance decisions on a relatively large section of roads can be a nontrivial task, due to various factors that need to be considered in the calculation. Even in a small area, different sections of a road can be in varying conditions. More importantly, road damages are under continuous development and changes. Decision-makers must be able to identify the critical sections, predict the temporal deterioration of every section of the roads, and ultimately distribute limited resources in a holistic way to optimize the long-term performance of the entire system, instead of local sections. As a result, fast detection and notification of undesired road conditions (such as potholes and rutting) is in a pressing need in Region 6 to improve the durability, sustainability, resilience of the critical road infrastructure, and ultimately assist the government efforts in infrastructure renovation and improve driving comfort and safety for the general public. It requires large amounts of high-quality live data about the road pavement conditions. Specifically, we must address the following technical difficulties:

1. How to collect high fidelity data of road pavement conditions in near real-time at the minimum cost (by leveraging existing technologies and hardware)?
2. How to enable the precise and reliable detection of the major road damages using technologies that are easily accessible for the public?
3. Following point 2, if the new technologies for road damage inspection are accessible for the public, how to leverage the concept of *crowdsourcing* to enable a participatory road inspection?
4. Finally, how to validate and test the road damage inspection results based on the crowdsourced data?

To address the technical challenges about intelligent and economic road inspection, this study aims to test an automated, crowdsourcing based approach that predicts road conditions based on running vehicles' vibration data via sensors built in most smartphones. A cloud-based smartphone app was developed to deploy in regular vehicles to enable participatory crowdsourcing in road damage data collection. We recognize that the vehicle vibration data as an instrument for road damage detection is challenged by the low quality and variability of the collected data, as it can be affected by the different models, ages, conditions of the vehicles, as well as the varying driving behaviors of the drivers. As a result, we tested the use of the latest advances in Deep Learning methods, including Sparse Coding and Self-Taught Learning (STL) to deal with data quality issues. Evidence shows that Deep Learning is robust to low-quality (but potentially large amount) data. Specifically, the

STL method has been proven to be effective when there is no enough labeled data for model training. The remainder of the report will introduce the background, methodology, and analysis results of the proposed method.

2. OBJECTIVES

The main objective of this research is to develop a participatory, economical and intelligent method that automatically predicts road damages based on the vibration patterns of vehicles running on the road, via sensors built in most smartphones of the daily drivers. By leveraging the power of numerous regular vehicles and Deep Learning technologies, the outcome of this research project will help optimize the maintenance decisions in Region 6, and significantly improve the durability, sustainability, and resilience of the roads in Region 6. The technical objectives include:

Objective 1: Develop and test a vehicle vibration-based road damage data collection method using smartphones

This research aims to leverage millions of regular vehicles to provide a significant amount of raw data that implies the rough information about road conditions. Vehicle vibration is closely related to the road condition and thus is a valuable data source for road damage detection. In order to reduce the technology cost, the vibration data collection was done with the personal smartphones of drivers. Most smartphones have built-in high-accuracy accelerometers and gyroscope sensors that can help recover the vibration patterns of the vehicles. Objective 1 will leverage the use of smartphones to collect road damage data, including imagery data of road damages and vehicle vibrations, for the model training. It also involves the labeling of specific road damages based on the imagery data to build the ground truth for model training.

Objective 2: Model the relationship between vehicle vibrations and the ground truth of major road damages

Classic vibration data classification methods rely on features extracted from the frequency domain, such as Spectrum Analysis. The challenge pertaining to the vehicle vibration data is the obvious noise contained in data– for example, different brands and conditions of vehicles, different weathers, different traffic conditions, and varying diving behaviors can all affect the quality and consistency of the collected data. As a result, the second objective is to analyze the data using the latest deep learning technologies, such as Sparse Coding. Evidence has shown that deep learning is very good at processing low-quality data if the data size is sufficient *(10)*. The quality of vehicle vibration data is low in terms of telling the precise road conditions, but the amount of data can be dramatically attributed to the big number of vehicles running on roads. As a result, the noise contained in the raw data can be filtered out by leveraging the power of deep learning technologies and the big data size.

Objective 3: Validate and test the road damage detection model enabled by vehicle vibration patterns

Finally, we aim to collect direct evidence about the effectiveness and efficiency of the proposed vehicle vibration-based road inspection method in comparison with the existing methods. We will also discuss the methods of encouraging broader participation of the public in the crowdsourcing of road inspection data. In order to enable the participation of a broader population, we developed a Cloud-based smartphone app that has the following features: 1) it does not require users to download and install the app; instead, users can use the built-in apps, such as web browsers available in most of the smartphones, to access the app. To achieve this objective, we will test the use of JavaScript that is characterized as dynamic, weakly typed, prototype-based and multi-paradigm; and 2) it allows the users to upload and download collected data to a centralized cloud server automatically. A centralized server will help us better manage the dataset and avoid potential data missing and duplicate.

3. LITERATURE REVIEW

3.1. Current Road Inspection Practices

At present, road surface inspection in region 6 is usually done with a manual process, such as shown in Figure 3 *(11)*. DOTs have published standard procedures for manual inspection *(12)*, such as Texas Pavement Management Information System Rater's Manual *(13)*. The manual approach brings two potential problems: variation in inspection results due to inspectors' personal bias, and the difficulty of high-frequent inspection and coverage area. To overcome the limitations of manual inspection, a variety of automated road surface inspection methods have been proposed *(14; 15)*. Representative technologies include vision-based method *(11; 16-19)*, laser scanning (*20*), ground penetration radar (GPR) *(11; 21)*, natural lighting method *(22)* and a combination of multiple sensors *(23)*. For example, Huang and Xu *(19)* developed a road surface inspection method based on the grid cell analysis of the collected grey images of the road surface. Zhang et al. *(24)* tested the use of deep Convolutional Neural Network (CNN) in categorizing pavement damages based on imagery data. Zhang et al. *(24)*, Ouyang and Xu *(25)*, and Laurent et al. *(26)* tested the use of car-mounted 3D laser scanners in automated road surface inspection.

Figure 3. The state of the art of road inspection – subjective manual inspection and expensive sensors, left: Manual road inspection is subjective, time-consuming, and disruptive, right: Sensor-based road inspection is more accurate but expensive and difficult to deploy.

However, these sensors-based technologies also have their limitations. Despite the accuracy and effectiveness of these sensing methods, they are usually costly, and thus the coverage and collection frequency can remain insufficient for detecting the dynamically changing road conditions. This project proposes to adopt a machine learning approach to predict road conditions based on running vehicles' vibration data, via sensors built in most smartphones. First, regular vehicles will be leveraged to provide a big amount of raw data that implies the rough information about road conditions. Vehicle vibration is closely related to the road condition and thus is a valuable data source for road condition prediction. The challenge pertains to the obvious noise contained in data– for example, different brands and conditions of vehicles, different weathers, different traffic conditions, and varying diving behaviors can all affect the quality and consistency of the collected data. As a result, the second step is to analyze the data using deep learning technologies, such as Sparse Coding. Evidence has shown that deep learning is very good at processing low-quality data if the data size is sufficient *(10)*. The quality of vehicle vibration data

is low in terms of telling the precise road conditions, but the amount of data can be dramatically attributed to the big number of participants. As a result, the noise contained in the raw data can be filtered out by leveraging the power of deep learning technologies.

3.2. International Roughness Index (IRI) Studies

In addition, many DOTs have widely adopted the International Roughness Index (IRI) in road inspection *(27-29)*. IRI is the roughness index most commonly obtained from measured longitudinal road profiles, by using a quarter-car vehicle math model, whose response is accumulated to yield a roughness index with units of slope (in/mi, m/km, etc.) *(30)*. Since its introduction in 1986, IRI has become an index that is most commonly used worldwide for evaluating and managing road systems. In the US, The measurement of IRI is required for data provided to the United States Federal Highway Administration (FHA), covered in several ASTM standards including ASTM E1926 – 08 *(31)*, ASTM E1364 - 95(2005) *(32)*, etc. IRI is usually obtained by a direct inspection to road profile. Recently, literature has begun the investigation into the use of vehicle vibrations in automated IRI calculation *(33-36)*. For example, Douangphachanh and Oneyama *(36)* developed and tested a car carried smartphone system that can identify the features and relationship of acceleration vibration that are useful in predicting the levels of IRI. This research contributes to the exiting, IRI literature by expanding the IRI level prediction into a high-fidelity road damage prediction. In our pilot study we found that vehicle vibration data has the potential to predict not only the level of IRI, but also the specific road damage types such as raveling, rutting, cracks and potholes etc. By leveraging the power of deep learning techniques, we expect to obtain semantically rich information about the specific road damages.

3.3. Vehicle Vibration for Road IRI Assessment

Literature has tested quarter-car vehicle models for assessing road conditions using IRI *(37)*. Roadroid *(38)* is an Android smartphone application that surveys road condition and classifies it as good, satisfactory, unsatisfactory, and poor based on calculated and estimated IRI. Li and Goldberg *(39)* calculated a proxy-IRI value that is linearly related to IRI. Although IRI is a common road roughness index measure worldwide, it sometimes fails to recognize isolated faults on smooth roads as it is calculated for a stretch of road using the road's profile. Lepine, Rouillard, and Sek *(40)* implemented different machine learning algorithms to separate non-stationary vibrations, and transient shocks present road vehicle vibration (RVV) signals using accelerometer data. He concludes that machine learning algorithms can be optimized and tuned to achieve high accuracy in detecting road vehicle vibration shocks. However, the road vehicle vibration signals he used were artificially generated using non-stationary random vibration and shock impulses that reproduce typical vehicle dynamic behavior. Allouch et al. *(41)* used machine learning techniques such as C4.5 Decision Tree, SVM, and Naïve Bayes to label road conditions as 'Smooth' or 'Potholed'. Bhoraskar et al. *(42)* used k-means clustering and SVM to label road conditions as 'Smooth' or 'Bumpy'. Silva et al. *(43)* approached the problem with data mining using Scikit-learn and Weka to detect unlevelled manholes, short bumps, and long bumps. Several other works use similar techniques and focus on either estimating a roughness metric or detecting potholes only *(44-47)*. However, effective road lifecycle management requires timely maintenance in stages prior to pothole formation such as cracking, shoving, delamination, etc. Crack detection using accelerometers is challenging due to its subtle vibration pattern and vehicle vibration noise. Several video image processing techniques have been suggested *(48; 49)*, but these techniques are

memory and computation intensive. With significant advancements in big data analytics and a push for smart cars in recent years, the high volume of driving data can be collected from users and processed to obtain useful information. Li and Goldberg *(39)* and Masino *(50)* proposed the use of crowd-sensing to obtain data and classify road conditions.

To summarize the literature review, there are certain areas that can be explored or improved for road damage detection using smartphone sensors. The majority of the current literature focuses on binary classifications using simple machine learning techniques or threshold-based heuristics to assess levels of IRI, or significant damages only, such as potholes. The multiclass classification has not been explored for different stages of road deterioration and different types of road damages. Data used for road damage detection only focus on vibration signals collected from acceleration data in the direction of gravity. Information and relationships between data that may be present in other directions orthogonal to the direction of gravity are not taken into consideration. Finally, the use of neural networks for multiclass classification has not been explored. Our research aims to analyze different machine learning techniques for multiclass classification to classify diverse road damages.

4. METHODOLOGY

This research developed a deep learning approach to enable real-time, economic, and automatic road condition inspection based on vehicle vibrations. Eventually, precise real-time road condition prediction can be achieved to support informed decision-making. Figure 4 illustrates the methodology of this research.

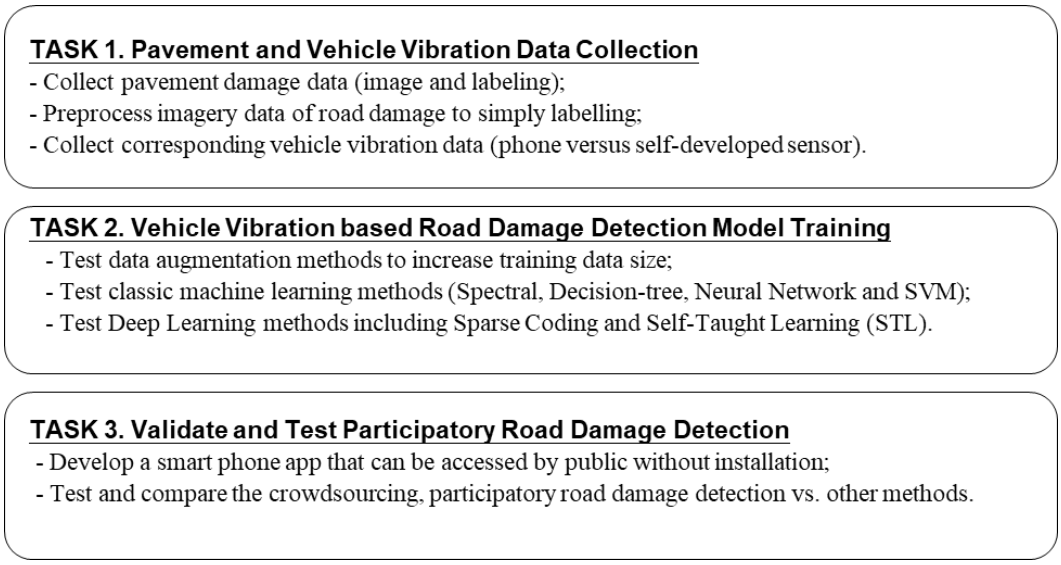

Figure 4. Research methodology.

The proposed method advances state of the art in three ways: **First**, it is economic. The proposed method adopts a crowdsourcing approach in automated road inspection tasks. Crowdsourcing is an emerging paradigm in computing that employs the power of human workers' knowledge and expertise to help solve problems that machines cannot solve alone *(51*). Recently, there has been increasing interest among researchers in employing crowdsourcing to tackle a wide range of complex engineering problems *(52-56*). Millions of regular vehicles, and personal smartphones provide the least expensive data collection solution. **Second**, the proposed method enables a 24/7, full coverage road inspection. As long as there are vehicles traveling on the road, the road condition data will be continuously fed into the central database for analysis. Moreover, unlike the manual inspection or sensor-based inspection, the coverage of data sampling is also outstanding. **Third**, the proposed method enables a longitude analysis that can help build a precise temporal prediction model of road deterioration. Since vehicle vibration-based road condition data can be collected seamlessly, it is possible to develop a longitudinal model (i.e., road condition change over time) based on the existing civil engineering knowledge and the big data analytics framework. By connecting the data points collected in a relatively long period of time (e.g., a year), engineers can project the trend of pavement distress development. Decision-makers and drivers can be notified of possible dangerous areas before the real damages happen. Government agencies can make preventive maintenance decisions that are much more proactive and effective. For end-users (government decision-makers and drivers), the proposed method has the great potential to provide an intelligent, user-friendly interface for road condition warnings.

The research team involves two faculty members from two institutions: Dr. Changbum Ahn (previously Dr. Jing Du) at Texas A&M University (TAMU), and Dr. Chao Wang at Louisiana State University (LSU). In addition, two graduate students from two institutions were supported

by this project to assistant in the research activities. To enhance the collaboration between two institutions, the PIs held virtual meetings on a weekly basis, and face-to-face meetings every semester. The graduate students supported by this project were co-advised by the PIs from two institutions. Meanwhile, an advisory board was established to monitor and evaluate the progress of the proposed research activities. The team created a shared cloud server to store files, documents, progress reports, presentations, etc. for all members.

5. ANALYSIS AND FINDINGS

5.1. Data Collection and Processing

Our first task was to collect road condition data (image and labeling) and the corresponding vehicle vibration data for model training.

5.1.1. Collect and Label Imagery Data of Road Damages

We installed a DJI Osmo Gimbal 12.76 MP Ultra HD Action Camera (4K) on the hook of a vehicle to collect the imagery data of pavement damages; the camera was mounted to the hook of a running vehicle (Figure 5) while pointing to the ground. Videos of the road surface were collected at the frame rate of 120 fps, and the resolution of 4K (4096*2160). The car was set to a cruise control mode with a constant speed of 30 miles per hour, and hence the preliminary analysis was more accurate. We collected at least 10 minutes of video on each of the selected road sections. It represents 5 miles of pavement imagery data.

Figure 5. Car mounted 4K camera.

Then we developed a program in Matlab to divide the 4K video into each frame for later labeling. To reduce the amount of data, we applied an image processing algorithm to highlight the frames with significant patterns or those highly susceptible to potential pavement damages. Figure 6 illustrates the image processing results of a few example of pavement damages that are beyond our preset threshold. The other frames that did not present significant patterns were associated with less concerned road sections and were discarded.

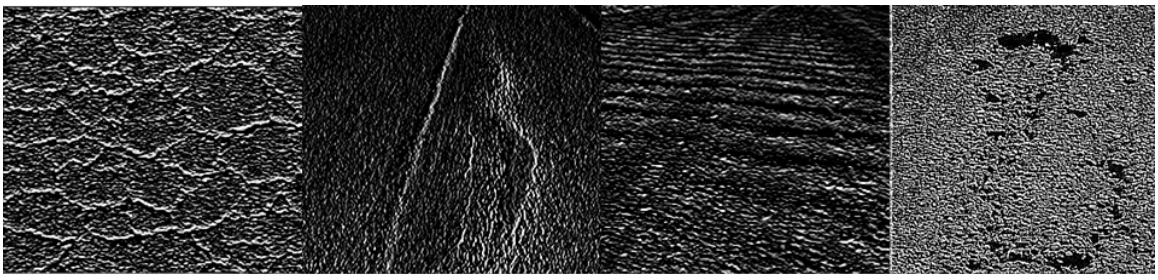

Figure 6. Image processing results of significant pavement damages beyond the set threshold.

Once we have the imagery data, we needed to label the pavement damages to establish the ground truth for model training. We shared the filtered images of pavement damages with 10 Construction Science graduate students at TAMU who were taking COSC 644 – Advanced Construction Systems and 10 construction management graduate students at LSU who were attending CM 7010 – Research Methods in Construction Management. The students were required to label the damages based on their best knowledge. The final label of each identified pavement damage was determined by the percentage of voting.

5.1.2. Collect Vehicle Vibration Data

We also collected the vehicle vibration data and synchronized it with the labeled pavement damages. To enable vibration data collection and labeling, we developed an iOS app that can record vehicle vibration in real-time at the frequency of 100 Hz (Figure 7a). It also records the current GPS location of the device and calculates the speed of the vehicle based on the change in GPS coordinates. The app registers precise UNIX timestamp of each data point for further analysis. Then, in order to train the model, we also collected 4K images of the road conditions, synchronized with the vibration data based on the UNIX timestamp. A DJI Osmo 4k Handheld camera with Gimbal was used to record the video of the road ahead to label the data collected (Figure 7b). The gimbal acted as a self-balancing mechanism to avoid motion blur as well as rotation/movement due to vibration and acceleration. The device is capable of recording videos at 4k, but for our purposes, the specs are set to record 720p videos at 60fps. The device was placed on the moving vehicle using the DJI OSMO 3 suction car mount shown in the pictures. The videos collected were used only for training purposes to label the training vibration data based on the road condition. Last, we developed an analysis program to predict road conditions based on the moving vehicle's vibration patterns (Figure 7c). We developed an iOS app for vehicle vibration collection and data synchronization.

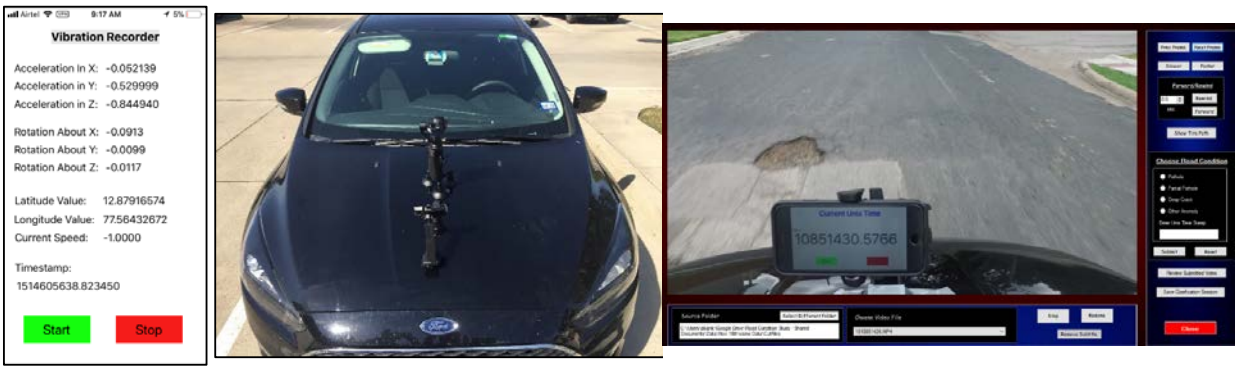

Figure 7. The developed iOS app and program for training data collection and labeling, left: Self-developed iOS app, middle: Vehicle and camera used in the experiments, right: Self-developed data analysis program.

We deployed the iOS app on 3 cars and collected 310 miles of road surface imagery data and corresponding vehicle vibration data in College Station, TX, and Baton Rouge, LA. We also developed a Google Map add-in to visualize the paths and vehicle vibration levels, as shown in Figure 8.

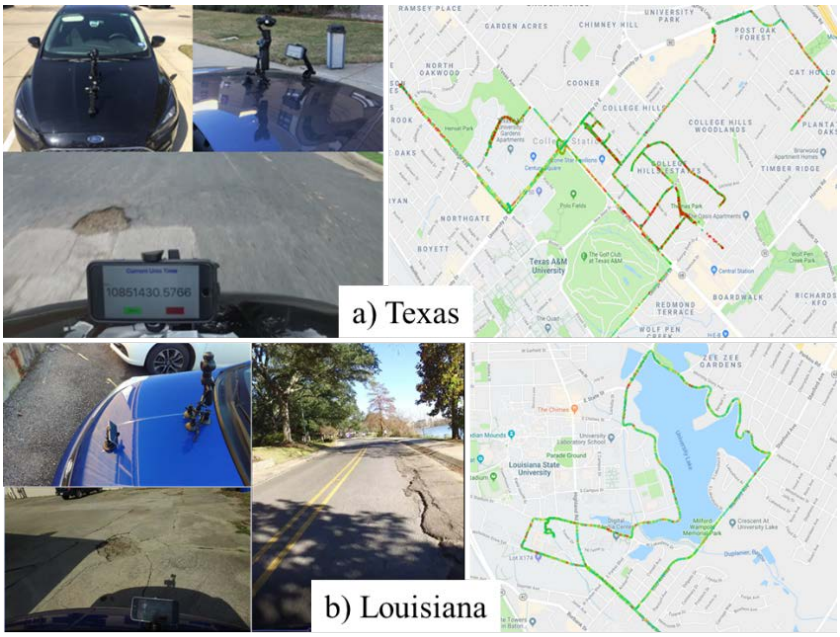

Figure 8. Data collection in both Texas and Louisiana.

5.1.3. Data Preprocessing

Data acquired were pre-processed in several stages to make it more coherent and pragmatic. First, the acceleration data collected needed to be virtually reoriented to a global frame of reference to remove variations due to the phone's position and orientation. The acceleration and gyroscope measurements are recorded in a three-dimensional Cartesian coordinate system concerning the phone's frame of reference, as shown in Figure 9. To maintain uniformity and integrity of the data collected from multiple data runs, the phone's frame of reference was transformed to a global frame of reference concerning the ground, as shown in Figure 10.

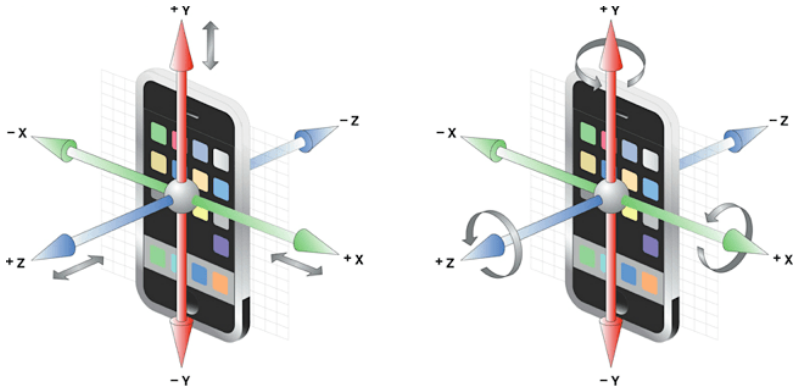

Figure 9. Cartesian coordinate axes of iPhone accelerometer and gyroscope.

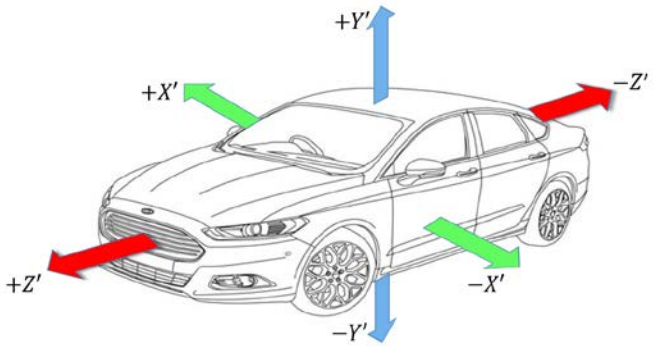

Figure 10. The global frame of reference: Cartesian coordinate axes w.r.t. ground.

The reorientation algorithm performs accelerometer data reorientation using Euler's angles, which form a representation of the spatial orientation of a certain reference frame as a combination of three orthogonal elemental rotations. Ideally, when a car is at rest on a flat surface, the acceleration values would be:

a_x=0m/s^2, a_y=9.81m/s^2 and a_z=0m/s^2

Equations [1] to [4] are used to calculate two of the three Euler angles and reorient acceleration values to the global frame of reference [30]. $a'_x, a'_y\ a'_z$ are the acceleration values with respect to the global reference frame while α and β are the roll and pitch angles, respectively. Figure 11 shows the plot of the acceleration data of a 1.5s window before and after reorientation.

$$\alpha = \tan^{-1}\left(\frac{a_y}{a_z}\right) \qquad \beta = \tan^{-1}\left(\frac{-a_x}{\sqrt{(a_y)^2+(a_z)^2}}\right) \qquad [1]$$

$$a'_x = \cos(\beta)\, a_x + \sin(\beta)\sin(\alpha)\, a_y + \cos(\alpha)\sin(\beta)\, a_z \qquad [2]$$

$$a'_y = \cos(\alpha)\, a_y - \sin(\alpha)\, a_z \qquad [3]$$

$$a'_z = -\sin(\beta)\, a_x + \cos(\beta)\sin(\alpha)\, a_y + \cos(\alpha)\cos(\beta)\, a_z \qquad [4]$$

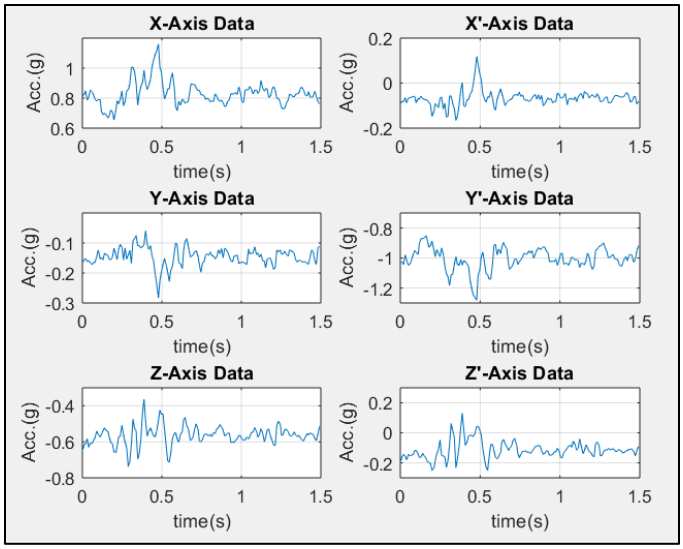

Figure 11. Reorientation of acceleration data to a global frame of reference.

The next stage of pre-processing requires the road surface condition to be labeled in order to obtain the ground truth for our supervised machine learning algorithms. Road pavement surface was classified as Potholes, Deep Transverse Cracks or Smooth Road by following guidelines and descriptions provided in pavement maintenance manuals from the Texas Department of Transportation (example shown in Figure 12). Transverse cracks that created a pavement elevation or depression of over 0.5 inches at the position of the crack were considered to be Deep Transverse Cracks.

Figure 12. Instances of road damages: Deep transverse crack and pothole.

A custom software application was developed to label the video data that was recorded. It enabled the user to perform standard video playback operations such as play, pause, fast-forward, rewind, and view frame-by-frame. Since our interest lies only in the section of the road that the car tires travel over, it also provided a feature to overlay the projected tire-trajectory onto the video frames as shown in Figure 13 Instances where the car tires partially travel over a road anomaly was labelled as an anomaly if it covered at least 60% of the tire width. Finally, the user assigns a label to the road segment by selecting a certain frame and specifying the anomaly, and the current timestamp displayed.

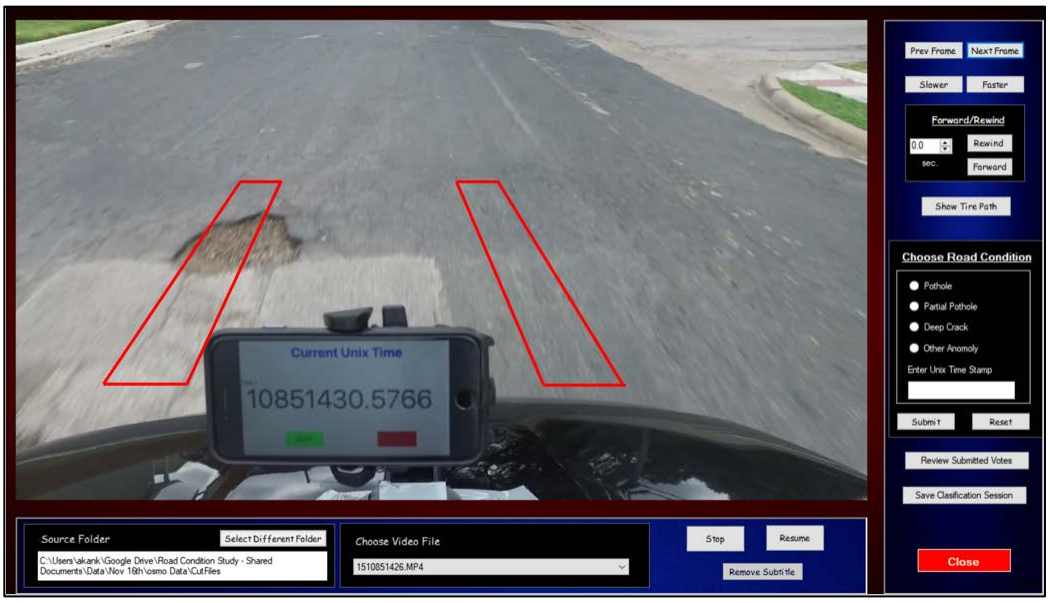

Figure 13. Road condition classifier software with tire-trajectory overlay.

Next, in order to geographically localize the instances of road conditions recorded, the recorded GPS data was synced with the vibration data collected using the timestamps. The speed of the vehicle was calculated based on the rate of change in GPS coordinates. However, due to the difference in the sampling rate of the Accelerometer/Gyroscope and the GPS sensor, the GPS data and the vehicle speed was interpolated using a spline transformation. This provided a reasonably accurate estimation of the location and speed at a higher sampling rate.

Furthermore, to remove certain driving conditions that are not related to the quality of road surface such as acceleration, stopping, braking, lane changing, turning, etc., the acceleration data in the X' and Z' axis was filtered with a Butterworth high-pass filter of order 11, cut-off frequency of 3Hz and attenuation of 80dB. The filter removes low-frequency components related to these events while preserving any high-frequency changes due to road anomalies, as shown in Figure 14. To analyze the information contained in higher frequency bands due to the anomalies, a low pass filter or smoothing filter was not applied.

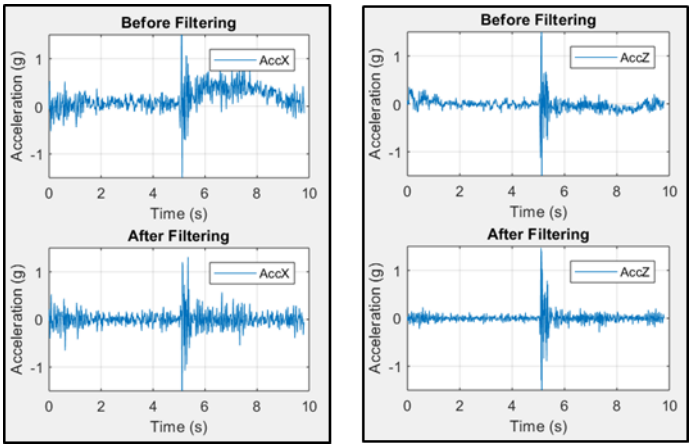

Figure 14. Acceleration signal in the X' and Z' axis before and after filtering.

The continuous filtered data was then converted into segments of data windows of length 100 data samples with a 50% overlap of windows. Labeled anomalies and smooth road segments were extracted and stored separately for further processing described in the feature extraction section. From all data collected, a dataset of 209 window segments was taken into consideration which contained 155 pothole instances, 34 deep crack instances, and 20 bump instances window segments.

5.1.4. Data Augmentation

A problem we met was the lack of high-quality training data for the model. Especially at this point, we have only identified ~200 locations in TX and LA with main road pavement damages. As a result, we tested the use of data augmentation to expand the effective database. Data augmentation methods, i.e., a set of techniques for generating artificial data through the expansion of an original dataset(*57*). The augmenters, such as jittering, scaling, magnitude warping, and time warping, help expand the dataset through probabilistic processing while still preserving the main features. Here we present the following examples:

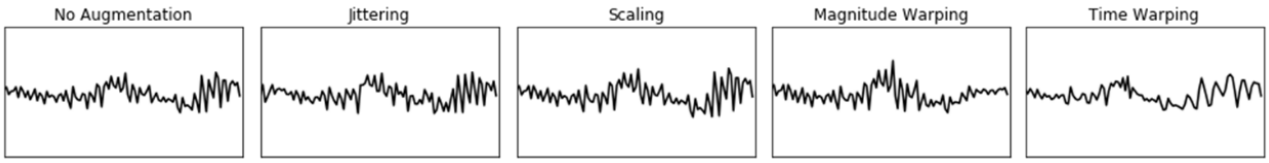

Figure 15. Methods of data augmentation.

Jittering simulates additive sensor noise. Scaling changes the magnitude of the data in a window by multiplying a random scale. Magnitude changes the magnitude of each sample by convolving the data window with a smooth curve varying around one. Time-warping distorts the time intervals between samples.

```python
import numpy as np
from scipy.interpolate import CubicSpline

def jitter(X, sigma=0.1):
    """
    sigma: standard devitation (STD) of the noise
    """
    myNoise = np.random.normal(loc=0, scale=sigma, size=X.shape)
    return X+myNoise

def scaling(X, sigma=0.1):
    """
    sigma: STD of the zoom-in/out factor
    """
    scalingFactor = np.random.normal(loc=1.0, scale=sigma, size=(1, X.shape[1]))
    myNoise = np.matmul(np.ones((X.shape[0],1)), scalingFactor)
    return X*myNoise

def generateRandomCurves(X, sigma=0.2, knot=4):
    """
    sigma: STD of the random knots for generating curves
    knot: # of knots for the random curves (complexity of the curves)
    """
    xx = (np.ones((X.shape[1],1))*(np.arange(0,X.shape[0], (X.shape[0]-1)/(knot+1)))).transpose()
    yy = np.random.normal(loc=1.0, scale=sigma, size=(knot+2, X.shape[1]))
    x_range = np.arange(X.shape[0])
    cs_x = CubicSpline(xx[:,0], yy[:,0])
    cs_y = CubicSpline(xx[:,1], yy[:,1])
    cs_z = CubicSpline(xx[:,2], yy[:,2])
    return np.array([cs_x(x_range),cs_y(x_range),cs_z(x_range)]).transpose()

def magWarp(X, sigma=0.1):
    return X * self.generateRandomCurves(X, sigma)

def distortTimesteps(X, sigma=0.2):
    tt = self.generateRandomCurves(X, sigma)
    tt_cum = np.cumsum(tt, axis=0)
    t_scale = [(X.shape[0]-1)/tt_cum[-1,0],(X.shape[0]-1)/tt_cum[-1,1],(X.shape[0]-1)/tt_cum[-1,2]]
    tt_cum[:,0] = tt_cum[:,0]*t_scale[0]
    tt_cum[:,1] = tt_cum[:,1]*t_scale[1]
    tt_cum[:,2] = tt_cum[:,2]*t_scale[2]
    return tt_cum

def timeWarp(X, sigma=0.2):
    tt_new = self.distortTimesteps(X, sigma)
    X_new = np.zeros(X.shape)
    x_range = np.arange(X.shape[0])
    X_new[:,0] = np.interp(x_range, tt_new[:,0], X[:,0])
    X_new[:,1] = np.interp(x_range, tt_new[:,1], X[:,1])
    X_new[:,2] = np.interp(x_range, tt_new[:,2], X[:,2])
    return X_new
```

Figure 16. Sample codes for data augmentation.

5.2. Model Training – Traditional Machine Learning Methods

5.2.1. Feature Extraction

Previous works in literature only used a few selected features that were considered to provide good distinction between road conditions. However, we wanted to comprehensively explore various possible features to extract any useful information provided by them. After reviewing various possible parameters mentioned by Gadelmawla et al. and previous literature, various time-domain

measures such as Maximum Value, Minimum Value, Mean Value, RMS Value, Peak-to-Peak Value and Ten-Point Average Value were calculated from the time domain signal, its peaks, troughs, and signal envelopes.

In the frequency domain, the power spectral density of vibration signals provides beneficial information that could be used to distinguish different road conditions. The power spectral density was calculated for the windowed signals, and the entire bandwidth was divided into smaller bands of 5Hz each. For each of these bands, average band power, RMS band value, and maximum band value were considered as frequency-domain features.

In the wavelet domain, Mortlet wavelets and Daubechies wavelets were deemed suitable to analyze vibration patterns due to road conditions. We conducted a study to determine suitable mother wavelets by comparing Haar, Mortlet, Mexican Hat, and Daubechies 6 and 10. She concluded that the Mortlet wavelet, as well as Daubechies 6 and 10 wavelets, could be used to effectively analyze road vehicle vibrations. Upon preliminary study, scales 4 and 5 for each of the three wavelets showed the most distinguishable characteristics for different road conditions. RMS values and ten-point averages of these scales were considered as wavelet domain features.

In previous literature, as mentioned in the introduction section, acceleration in the Y' direction was considered to contain most of the features needed to adequately distinguish road anomalies. Accelerations in X' and Z' directions were considered for driving events only. However, we believe that more information regarding road anomalies are present in the X' and Z' directions. For example, when a car hits a pothole with its left front wheel, there is a sudden deceleration in the Z' direction as well as a sudden tilt in the X' direction. Such information may contribute to distinguishing cracks and potholes, considering cracks tend to span the entire width of the road whereas potholes are more localized. In total, 54 features were extracted from the accelerometer data for each of the three axes. Hence, each feature vector consisted of 162 feature values that were saved as a .MAT file.

5.2.2. Overview of Classic Machine Learning Methods

Machine Learning is an application of Artificial Intelligence (AI) that provides computer systems the ability to learn and improve from experience without explicit programming. Once a computer algorithm is trained, the algorithm can apply the relationship learned during training to solve similar problems. For example, the retail industry, such as Amazon utilizes machine learning algorithms to provide highly personalized services. Data collected from prior purchases or searches are used as training data to classify online recommendations to specific users. This type of machine learning that divides analyzed data into discrete clusters or classes is referred to as the "classification problem". Another kind of machine learning problem, known as "regression problem", finds continuous relationships between data variables instead of clustering data into different classes.

Figure 17 shows the general workflow for both classification and regression type of machine learning approaches. It begins with a dataset of raw data whose class labels are previously known. For the case of road vehicle vibration, this is the acceleration signals recorded by the smartphone which are labeled with different road conditions. This input dataset is processed to obtain various attributes of the data called features that are compatible with machine learning algorithms. Once

the features and class labels are extracted, the features list and corresponding class labels are partitioned into three sets, the training dataset, validation dataset, and testing dataset. All three sets have the same distribution of classes in terms of proportion. The training set is used to train the algorithm and develop the classifier model. The validation dataset is then used to validate the performance of the trained classifier. If there is not enough data to create a validation set, there are several other approaches for validation of models such as cross-validation where the entire data is used for both training and validation. The validation phase is useful to compare and correlate the performance of different models and choose the best one that fits the problem. To test the model on new data, the testing dataset is used as input to the final model to predict output data labels.

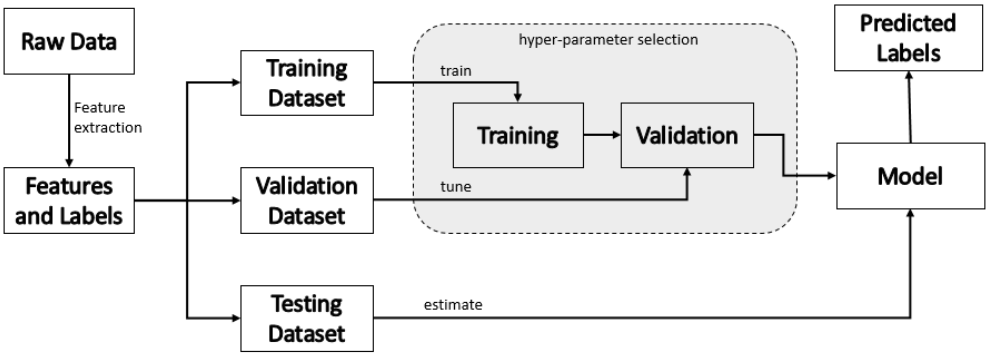

Figure 17. General workflow diagram of machine learning algorithms.

Various machine learning classification algorithms have been developed, which makes the selection of a classifier a problematic task. Since there is no standardized nomenclature in machine learning, similar classification algorithms may be expressed with different names. MATLAB® incorporates the Statistics and Machine Learning Toolbox, which included implementations of various machine learning classifiers. These classifiers can be primarily divided into seven categories- Naive Bayes Classification, Discriminant Analysis, Ensembles, Decision Trees, Nearest Neighbors, Support Vector Machines (SVM), and Neural Networks. For our study, SVM, Decision Trees, and Neural Networks were chosen as they are popular and reliable techniques used for classification of road vibration data.

The complete dataset is randomized and divided into training and testing dataset with an 80:20 ratio, keeping the proportion of the classes in both datasets constant. To investigate whether the models trained with input features extracted from all three axes perform better than using features from Y' axis only, two datasets containing 162 features and 54 features were created for each case respectively. The parameters used to analyze and quantify results are discussed in the Model Evaluation Parameters section.

5.2.3. Support Vector Machines (SVM)

Support Vector Machine is a supervised machine learning model that evaluates input data and recognizes patterns for classification and regression analysis. SVM performs classification by finding the hyperplane that maximizes the margin between data point clusters corresponding to different classes. SVMs are versatile, memory-efficient, and effective in high-dimensional spaces. Generally, SVM is used to classify data that have two distinct labels. In order to execute multi-

class SVM, MATLAB® incorporated the 'ClassificationECOC' class in their Statistics and Machine Learning Toolbox. ClassificationECOC is an Error-Correcting Output Code (ECOC) classifier used to perform multiclass learning by reducing the classifier to simple binary classifiers such as SVMs. An ECOC model reduces a classification problem involving at least three classes into a set of binary classifiers. If M is the coding design matrix with elements m_{kl} and s_l is the predicted classification score for the positive class of learner l, a new observation is assigned to the class $\hat{k}$ hat minimizes the aggregation of losses for the L binary learners given by Equation 5.

$$\hat{k} = \underset{k}{\text{again}} \frac{\sum_{l=1}^{L} |m_{kl}| g(m_{kl}, s_l)}{\sum_{l=1}^{L} |m_{kl}|} \qquad [5]$$

For this study, SVM was implemented in two ways- the Simple SVM and Cross Validated SVM. The Simple SVM implementation uses the default SVM binary learners and one-versus-one coding design to train the SVM model. However, this type of model tends to have the problem of over-fitting. In order to try and overcome this problem, a subset of data called a validation set is used to test the model during the training phase. Cross-validation techniques such as 5-fold cross-validation, 7-fold cross-validation, 10-fold cross-validation, and Leave One Out cross-validation are implemented for our analysis.

5.2.4. Decision Tree

Decision tree, also known as classification trees and regression trees, predict output responses based on input data. Following the decisions in the tree from the root node to the leaf node gives the output response to that particular input data. The decision tree is an algorithm that classifies data through a cascade of statistical tests, as shown in Figure 18. These tests compare the value that is input to a node with a threshold value that splits the tree's path. Tests can have multiple results, and different tree paths can follow to the same output class label. The complexity of the tree is defined by the number of branch splits and depending on its complexity; they have quick training and prediction speeds, moderate predictive accuracy, and low computational memory requirements.

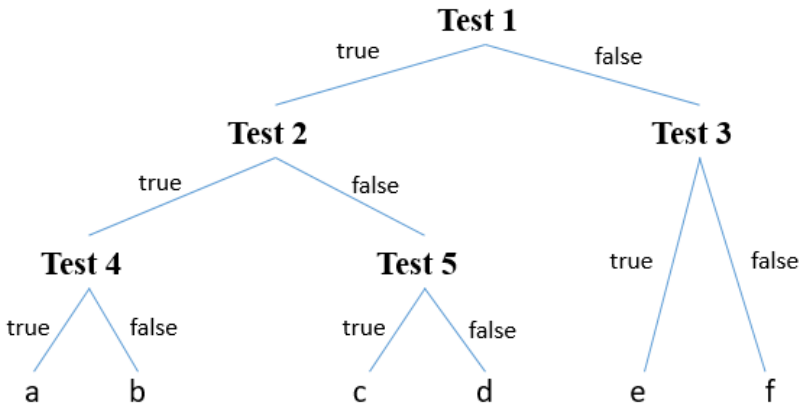

Figure 18. Decision tree structure.

The MATLAB® Statistics and Machine Learning Toolbox were used to train a binary classification decision tree for multiclass classification. Similar to our approach to SVM, we develop a simple classification decision tree, and a cross-validated tree to reduce over-fitting.

5.2.5. Neural Networks

Neural networks are a popular machine learning framework that attempts to imitate the learning pattern of natural biological neural networks in the brain. A typical neural network consists of inter-connected arithmetic processors called neurons which produce a sequence of real-valued activation outputs. Neurons present in the input layer of the neural network gets activated through sensor data perceiving the environment, while neurons present in other layers get activated through weighted connections from previously active neurons. Neural network algorithms link the feature vectors (input layer) to the class labels (output layer) using multilayered networks called hidden layers, as shown in Figure 19. The complexity of the classification problem determines the number of hidden layers needed. Although neural networks are powerful, high accuracy algorithms, training them requires a large dataset. The size of the required dataset also increases as the number of hidden layers increases.

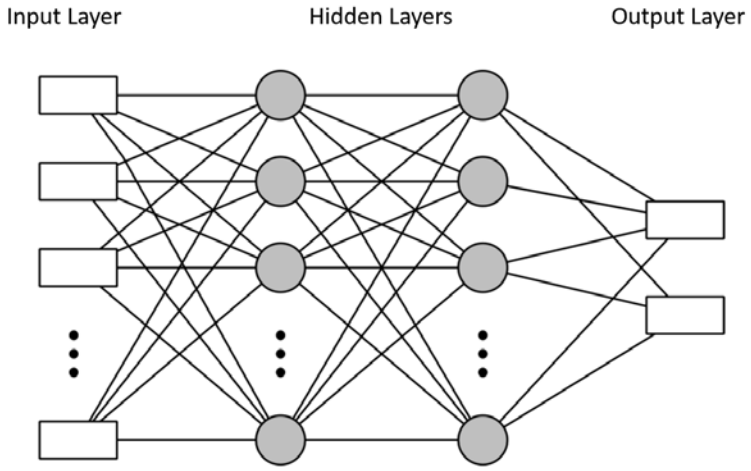

Figure 19. Structure of a Neural Network classifier with two hidden layers.

A multilayer perceptron (MLP) is a class of feedforward neural networks that comprises at least one hidden layer and uses backpropagation for training its models. Each neuron in the hidden layers uses a nonlinear activation function which distinguishes it from a linear perceptron. Each neuron inputs values from neurons in the previous layer and outputs the result of a weighted linear summation followed by a non-linear activation function. The output layer receives the values from the final hidden layer and outputs the class that is predicted for that input data.

To realize MLP networks, the scikit-learn library for supervised Neural Network was used. The training data and the testing data goes through additional pre-processing where the features are standardized by removing the mean and scaling to unit variance. This standardization step is a common requirement for various machine learning algorithms, including MLPs as they may perform poorly if the individual features do not resemble a standard normal distribution. The MLPClassifier class available in scikit-learn creates a model that optimizes the log-loss function using LBFGS or stochastic gradient descent. It includes various parameters such as activation function, hidden layer size, weight optimization solver, regularization factor, weight update learning rate, etc. to tune the model to the specific problem. After evaluating the performance of the classifier for all permutations of parameters, the MLP classifier that provided reliable results with high accuracy consisted of 7 to 10 hidden layers, an LBFGS weight optimization solver, a

constant learning rate for weight update and an activation function of ‘Tanh’ or ‘ReLU’.

LBFGS is a limited memory optimizer in the family of quasi-Newton methods that approximates the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm. For MLPs, the LBFGS solver can converge faster and performs well when dealing with small datasets. A comparison of activation functions ReLU, Tanh, and logistic sigmoid discussed in the Results Section showed that ReLU and Tanh perform better.

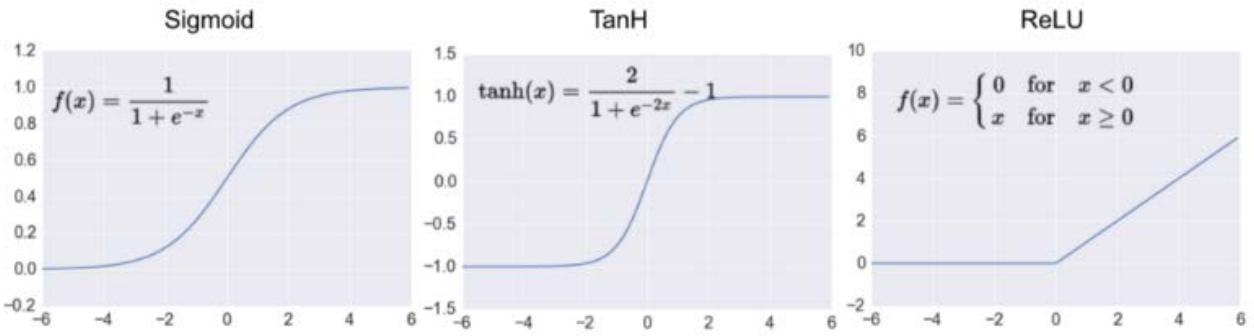

Figure 20. Activation function plots for Sigmoid, Tanh, and ReLU.

Since deep neural networks can be used with raw data and perform feature extraction implicitly, similar MLP classifiers were designed by providing the raw acceleration data as the input instead of the extracted features. As a window size of 100 data points was considered, each input vector had a length of 100 for the single Y' axis and 300 when all three axes were considered. Providing direct data to a neural network eliminates the process of manual feature extraction and hence saves time and memory in the training stage. However, such networks require a huge dataset in order to extract useful features and may not give high accuracy for the limited dataset we possess. Therefore, we not only explore the use of neural network classifiers in classifying feature vectors but also classifiers that can classify raw data directly. The results are provided in the Results section using the performance evaluation parameters discussed in the next section.

5.2.6. Model Evaluation

Evaluation Metrics: To evaluate the performance of the classifiers described in the previous section, various performance evaluation metrics are used for machine learning models. For each of the classifiers, we consider relevant and essential parameters which best enable us to derive a conclusion on its performance.

A confusion matrix is a specific tabular representation of the performance of a supervised machine learning algorithm. Each column represents the number of instances of the predicted class while each row represents the number of instances of an actual class. Most classification metrics are derived from the confusion matrix based on the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). A classifier’s accuracy, precision, and recall are described below:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

$$Precision = \frac{TP}{(TP + FP)}$$
$$Recall = \frac{TP}{(TP + FN)}$$

To evaluate the performance of the Simple SVM and Simple Decision Tree classifier, the average training loss, and average test accuracy for the trained classifier are recorded. The average training loss is the average in-sample loss of the trained classifier model using the training dataset while the average test accuracy is the average classification accuracy using the testing dataset for *n* iterations. The average precision and average recall for the three distinct classes predicted by the model are also recorded to analyze what proportion of identifications were correct and what proportion of actual positives were correctly identified. For cross-validated SVM and cross-validated Decision Tree, average training loss, and cross-validation error rates for k-fold, and leave-p-out cross-validations are recorded. Graphs of these parameters give an intuitive understanding of their reproducibility. Similarly, for the MLP classifier, the average training accuracy and average test accuracy is recorded for each of the selected combinations of parameters. This provides an overview of the classifier's performance while the Precision and Recall rates for each of the three classes provide more specific insights into the classifier's performance. We analyzed the obtained results and evaluate each of the machine learning model's capability for detecting road anomalies. The parameters used to measure and quantify performance are described in the previous section. The algorithms were run on an HP ENVY x360 convertible notebook running on Microsoft Windows 10 Home OS with an Intel® core™ i5-6200 processor, 2.30GHz CPU, and 8GB RAM. SVM and Decision tree algorithms were implemented using the Statistics and Machine Learning Toolbox on MATLAB® 2017, while Neural Network MLP implementation was carried out using Scikit-learn on Python 3.6 environment.

SVM Performance: The Simple SVM was implemented with one hundred iterations, each using distinct combinations of instances for the training and testing datasets while maintaining the same proportion of classes. Average values of evaluation parameters for these iterations were considered to evaluate the generalized performance of the algorithm. As discussed earlier, the SVM was trained separately using features from all three axes as well as features from only Y' axis to conduct a comparative analysis of performance. The simple SVM models trained were one-vs-one classifiers with equal misclassification cost and a linear kernel function. The training loss, testing accuracy, precision, and recall rates are tabulated in Table 1. The precision and recall rates are displayed for each of the three classes to analyze bias.

Table 1. Simple SVM implementation results.

Parameter	**All Axes**			**Y' Axis Only**		
Avg. Training Loss	**0.0279**			0.0773		
Avg. Test Accuracy	0.8855			**0.9015**		
	Crack	**Pothole**	**Smooth**	**Crack**	**Pothole**	**Smooth**
Avg. Precision	0.4025	0.7221	0.9442	0.3862	0.7479	0.9417
Avg. Recall	0.4375	0.6776	0.9471	0.2100	0.6568	0.9823

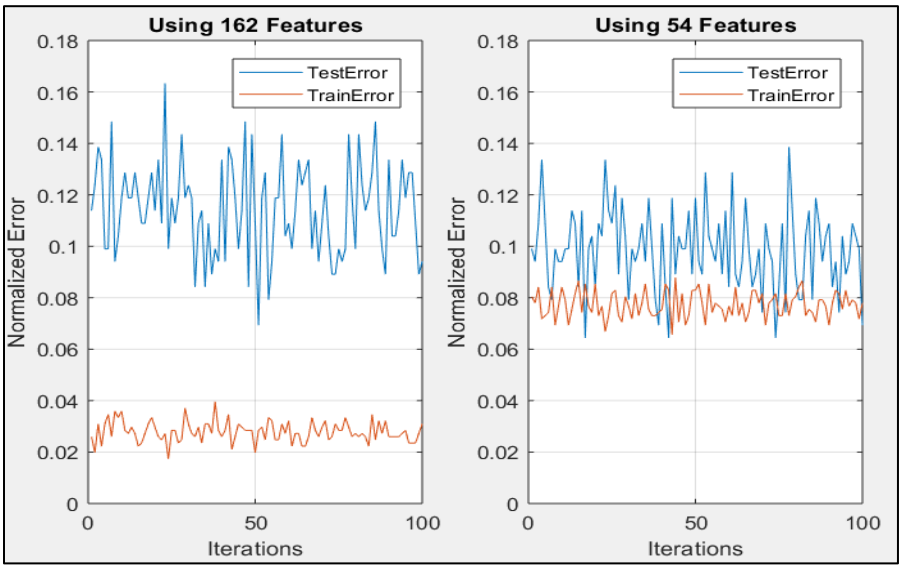

Figure 21. Simple SVM: Training and testing error rates using 162 features from 3 axes and 54 features from 1 axis.

The cross-validated SVM model also implements a one-vs-one classifier with a linear kernel function and measures performance using different cross-validation methods. The results of the cross-validation ECOC classifier for SVM is tabulated in Table 2.

Table 2. Cross validated SVM implementation results.

Parameter	All Axes	Y' Axis Only
Avg. Training Loss	**0.0149**	0.0663
Avg. 5-fold Loss	**0.0822**	0.0990
Avg. 7-fold Loss	**0.0851**	0.0990
Avg. 10-fold Loss	**0.0842**	0.0941
Avg. Leave One Out Loss	**0.0812**	0.0931

From Table 1, it is observed that the classifier trained with features from all three axes has a lower loss and performs much better than the classifier trained with features from Y' axis only. The average training loss is lower, and the average testing accuracy is higher for the former case. The precision and recall rates for the individual classes are also higher when all three axes are used. The recall rate for cracks shows the most significant improvement, going up by over 20%, while the recall for smooth road reduces by about 3.5%. The precision and recall rates for potholes remain very comparable. Table 2 shows that the cross-validated classifier with features from all three axes has a lower training loss, and lower cross-validated errors as well.

Decision Tree Performance: The decision trees are implemented similarly to SVM, with five hundred iterations of the simple decision tree being implemented with unique sets of training and testing data for each iteration. Decision trees are faster to train. However, they create a highly varying set of hyperparameters with each iteration such as a number of nodes and node thresholds. There exists a tradeoff between speed and reproducibility. The training loss, testing accuracy, precision, and recall rates of the simple decision tree implementation are tabulated in Table 3. The results of the cross-validated ECOC classifier for Decision Tree is tabulated in Table 4.

Table 3. Simple decision tree implementation results.

Parameter	All Axes			Y' Axis Only		
Avg. Training Loss	**0.0199**			0.0248		
Avg. Test Accuracy	**0.8835**			0.8734		
	Crack	**Pothole**	**Smooth**	**Crack**	**Pothole**	**Smooth**
Avg. Precision	0.4348	0.6663	0.9497	0.2925	0.6581	0.9442
Avg. Recall	0.4121	0.6716	0.9470	0.3080	0.6462	0.9471

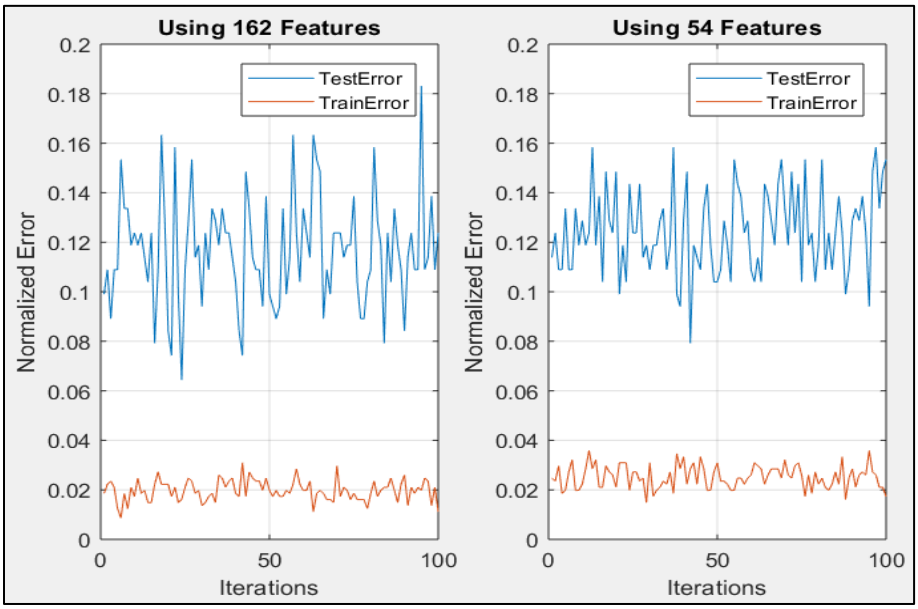

Figure 22. Simple decision tree: Training and testing error rates using 162 features from 3 axes and 54 features using 1 axis.

Table 4. Cross validated decision tree implementation results.

Parameter	All Axes	Y' Axis Only
Avg. Training Loss	**0.0188**	0.0267
Avg. 5-fold Loss	**0.1178**	0.1257
Avg. 7-fold Loss	0.1208	**0.1109**
Avg. 10-fold Loss	**0.1010**	0.1218
Avg. Leave One Out Loss	**0.0970**	0.1317

From Table 3, it is observed again that the classifiers trained with features from all three axes perform better than the classifiers trained with only Y' axis. Precision and Recall for cracks increase by over 10% each, and training loss, and testing accuracy shows slight improvements. However, from Figure 20, it is observed that the test accuracy is not very consistent across different iterations. This is expected as each iteration is trained well to a set of training data and may not perform as well with the testing data. Table 4 shows that the cross-validated classifier with all axes also performs better and shows lower training loss and cross-validation errors. However, when compared to SVM performance, the cross-validation errors are higher.

Neural Network Performance: The preliminary analysis stage of implementing an MLP neural network classifier involved a comparison of test accuracy, precision, and recall for the various combinations of parameters that were chosen. Twenty iterations of each set of parameters were implemented, and on inspection of the output performance metrics, the following conclusions were made: classifiers that implemented the Adam weight optimization solver gave a slightly better overall test accuracy than the LBFGS when used with activation function ReLU and comparable accuracy when used with activation function Tanh. However, the individual precision and recall rates for Crack and Pothole were much lower for Adam as compared to LBFGS. Classifiers that implemented LBFGS converged faster than Adam when the number of hidden layers was small but increased as the neural network grows deeper with more hidden layers. Comparison of precision and recall rates showed that the Tanh activation function gave poor precision and recall for cracks which were compensated in overall accuracy by high precision and recall for the smooth road. After the analysis, it was concluded that a classifier that implements LBFGS solver and hidden layer size 7 and 8 gave the most optimal results. However, there was a trade-off existed between ReLU, which yielded better precision for cracks and Tanh, which yielded better precision for the smooth road but gave very poor precision rates for cracks.

The final analysis stage compared the performance of the MLP neural networks for input feature vector lengths 162 and 54 while implementing ReLU and Tanh with LBFGS. The test accuracy, precision, and recall for these models are tabulated in Table 5 and Table 6.

Table 5. MLP implementation using ReLU- results.

MLP Hidden Layer Count	**All Axes**			**Y' Axis Only**		
Avg. Test Accuracy						
7	**0.9212**			0.8921		
8	**0.9190**			0.8919		
Avg. Precision Rates						
	Crack	Pothole	Smooth	Crack	Pothole	Smooth
7	**0.559**	**0.769**	**0.969**	0.350	0.674	0.962
8	**0.550**	**0.769**	**0.964**	0.345	0.688	0.959
Avg. Recall Rates						
	Crack	Pothole	Smooth	Crack	Pothole	Smooth
7	**0.611**	**0.799**	**0.962**	0.342	0.723	0.953
8	**0.585**	**0.781**	**0.963**	0.365	0.708	0.952

Table 6. MLP implementation using Tanh- results.

MLP Hidden Layer Count	**All Axes**			**Y' Axis Only**		
Avg. Test Accuracy						
7	**0.9122**			0.8978		
8	**0.9149**			0.8950		
Avg. Precision Rates						
	Crack	Pothole	Smooth	Crack	Pothole	Smooth
7	**0.486**	**0.757**	**0.964**	0.395	0.705	0.961
8	**0.491**	**0.754**	**0.967**	0.364	0.708	0.958
Avg. Recall Rates						
	Crack	Pothole	Smooth	Crack	Pothole	Smooth
7	**0.498**	**0.774**	**0.959**	0.409	0.738	0.956
8	**0.529**	**0.782**	**0.958**	0.416	0.718	0.955

Based on Table 5 and Table 6 it can be observed that the average test accuracy, precision, and recall rates are higher for the MLP models using features from all three axes compared to only a single axis. Considering models trained using features from only one axis, using Tanh as the activation function yields higher precision and recall rates among the three classes. However, when considering features from all three axes, ReLU stands out in its high precision and recall rates for cracks. The precision and recall rates for pothole and smooth remains quite similar between the two activation functions.

In order to test the performance of the MLP Neural Network classifiers in classifying road vibration data without manually performing feature extraction prior to training, the acceleration data is directly used as the input to the neural network and is evaluated over 20 iterations. The single-axis input vector has length 100 and input vector with all axes has length 300 with data each ax concatenated end to end. An initial analysis regarding the choice of activation function showed that Tanh activation function failed to produce significant precision and recall rates for cracks. Therefore, only ReLU was considered for analyzing MLP classifiers using direct data. The average test accuracy, precision, and recall for direct data input using ReLU activation function is tabulated in Table 7.

Table 7. MLP using direct data for ReLU- results.

MLP Hidden Layer Count	All Axes			Y' Axis Only		
Avg. Test Accuracy						
7	0.8027			0.8157		
8	0.7946			0.8112		
Avg. Precision Rates						
	Crack	Pothole	Smooth	Crack	Pothole	Smooth
7	0.283	0.329	0.918	0.271	0.423	0.911
8	0.408	0.301	0.906	0.258	0.469	0.905
Avg. Recall Rates						
	Crack	Pothole	Smooth	Crack	Pothole	Smooth
7	0.139	0.672	0.912	0.156	0.607	0.911
8	0.142	0.673	0.919	0.141	0.621	0.921

It is observed that the average test accuracy of MLP models using direct data is lower compared to MLP models trained using extracted features as input. The average precision and recall rates for the case of cracks and potholes are also lower. However, it was already anticipated that training neural networks without features would require a large dataset, and we are limited by the size and composition of our data. The main advantage of using Neural Networks without feature extraction is the time saved in feature extraction when realizing real-time systems. Earlier, we saw that on average, the feature extraction requires approximately 70ms and 25ms for the case of 3 axes and 1 axis respectively. Table 8 shows the average time required to classify a single data window using data from all three axes for different trained machine learning algorithms discussed in this paper. Since each of the classifiers take classification times in the order of microseconds, using MLP with direct data as the input would save computation time for feature extraction. When realizing such a system in real-time, this saves significant computation time.

Table 8. Classifier performance: Testing time.

Classifier	Avg. Time to Classify One Window (µs)
SVM	29.372
Decision Tree	4.8032
MLP	36.0142
MLP (Direct Data)	72.078

5.3. Model Training – Deep Learning

5.3.1. Sparse Coding

Then we tested Sparse Coding to examine if it can further improve model performance. Sparse coding is a class of unsupervised methods that learn data patterns based on extracted overcomplete bases. Although initially designed for image recognition, sparse coding has recently been used in classifying vibration data. The vibration of the moving vehicles is the combination of engine vibration and response of pavement. The pavement damages could cause acute abnormal impulses in the vibration signals. It is assumed that the vibration patterns of the auto engine are consistent

during driving. The acute impulses are regarded as the most characteristic patterns of the pavement distresses, and inspecting the abnormal impulses is equivalent to inspecting the vibration patterns of pavement damage. Therefore, the atoms of vibration signals can be learned by capturing the abnormal impulses and the pavement conditions videos, used to manually label the class of damage. In our experiments, 30 atoms, each with a length of 100-point (1 second of data) were learned from the Y-axis vibration signals labeled as pavement distresses (Figure 23). The reason for choosing the length of 100-point is that pavement damage would result in an impulse lasting less than 100-point. The dictionary learning process can adapt the most characteristic patterns, i.e., the abnormal impulse, in the vibration signals with the 100-point length.

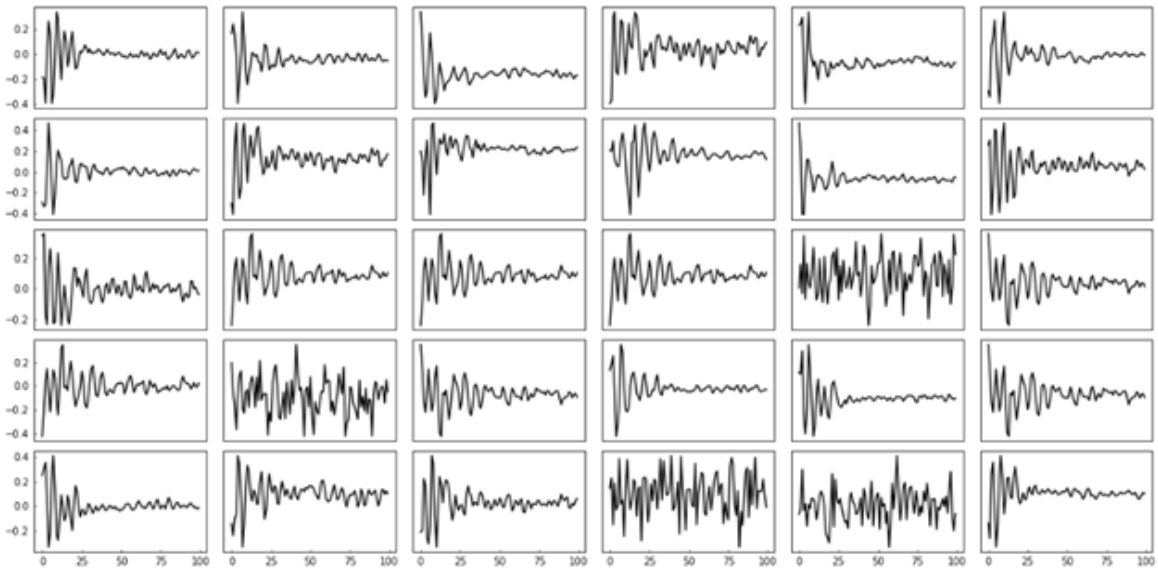

Figure 23. The learned atoms; atoms are the fundamental elements obtained from the sparse coding analysis that can be used to reproduce all raw vibration data.

Without loss of generality, three pavement damage conditions were tested respectively, including smooth pavement, big cracks, and potholes. We reconstructed the vibrations using the sparse representation, which is a linear combination of learned atoms, of the raw signals. The mean absolute reconstruction error of pothole, pothole part, and cracking are 0.11, 0.12, 0.11 amplitude, respectively. Figure 24 illustrates how Sparse Coding was used to reconstruct the temporal vibration data of a crack.

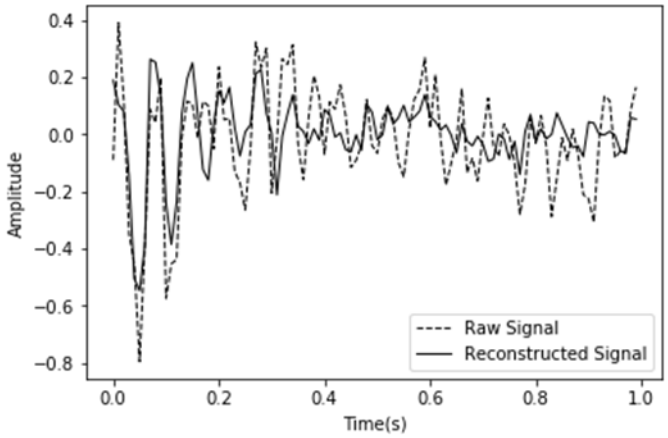

Figure 24. Comparison between a pothole Y-axis signal and its reconstruction based on atoms.

Then we performed the Sparse Dictionary Learning to find a dictionary that would adapt well to the raw data. The dictionary was applied with vibration data to solve the sparse representation of input signals, which would be utilized as sparse features for further classification. Figure 25 illustrates five sample signals and their corresponding sparse representations. It indicates that sparsity was evident for each signal's features, and vibrations signals could be represented as a sparse linear combination of atoms.

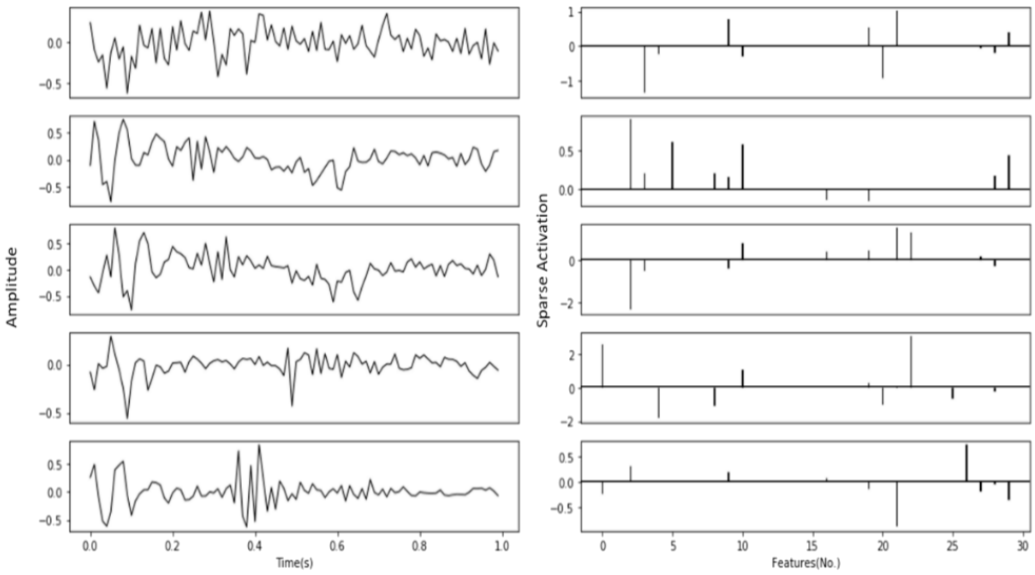

Figure 25. The original Y-axis signals and their corresponding sparse features, i.e., the linear combination coefficients of all atoms.

Finally, Support Vector Machines (SVMs) method was used to classify the road pavement damages. SVM is a supervised learning method for classification, regression, and outliners detection, which is defined by a separating hyperplane. Given labeled training data, the classifier would output a hyperplane which separates classes. SVM is effective in high dimensional spaces and versatile with a different kernel function.

5.3.2. Self-Taught Learning (STL)

To further improve classification performance, we also examined the latest advances in deep learning methods, including *Transfer Learning (TL)* and *Self-Taught Learning (STL)*, to make the best use of unlabeled data in model training. TL refers to a machine learning method that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem(*58*). For example, assume we do not have enough labeled data of driver ***i*** on a certain type of road damage but have a lot of labeled data of driver ***j*** on the same type of road damage. TL finds that user ***j***'s data can be used to train the model for user ***i***, because features should be logically relevant. STL, in contrast, takes a more aggressive approach by including unlabeled data of completely irrelevant classes(*59*), such as using driver ***i***'s data of damage ***a*** to train and predict driver ***j***'s data of damage b (note, damages are different too). According to increasing literature that supports STL(*59-62*), STL is effective in many cases possibly because different classes data, although seemingly totally irrelevant (like pothole induced vibrations vs. ravelling induced vibrations), may contain fundamentally similar basic patterns as they all follow the same physics rules. TL and STL have been widely applied in image classification but have not yet been widely tested in vibration learning. We proposed and tested the following algorithm sparse coding enabled STL:

Proposed Algorithm: STL via Sparse Coding

input Labeled training set
$T = \{(x_l^{(1)}, y^{(1)}), (x_l^{(2)}, y^{(2)}), \ldots, (x_l^{(m)}, y^{(m)})\}$.
Unlabeled data $\{x_u^{(1)}, x_u^{(2)}, \ldots, x_u^{(k)}\}$.

output Learned classifier for the classification task.

algorithm Using unlabeled data $\{x_u^{(i)}\}$, solve Sparse Coding problem to obtain bases b. Compute features for the classification task to obtain a new labeled training set $\hat{T} = \{(\hat{a}(x_l^{(i)}), y^{(i)})\}_{i=1}^m$, where $\hat{a}(x_l^{(i)}) = \arg\min_{a^{(i)}} \|x_l^{(i)} - \sum_j a_j^{(i)} b_j\|_2^2 + \beta \|a^{(i)}\|_1$. Learn a classifier $\mathcal{C}$ by applying a supervised learning algorithm (e.g., SVM) to the labeled training set $\hat{T}$.
return the learned classifier $\mathcal{C}$.

5.3.3. Deep Learning Results

In comparison with sparse features, Y-axis amplitude time sequences were used as time-domain features, that is, 200 amplitude features were selected for each signal with a length of 100-point. The dictionary with a set of atoms was learned using the minibatch dictionary learning in scikit-learn toolbox. Then apply the dictionary to the Y-axis vibration signals to extract the sparse features for each sample. Pavement damage recognition on these features was performed using SVM classifier found in the scikit-learn toolbox. Table 1 shows the recognition results for the SVM classifier. Overall, recognition is highest for a pothole, which was 76% and 80% based on time sequence features and sparse features, respectively. The sparse features approach has little improvement compared to time-domain method. The recognition accuracy is low for bump and cracking because of lack of data. The other reason why bump and cracking have low recognition rates and misclassified as a pothole is that bump and cracking both involve the similar vibration pattern of the pothole. In particular, it was found that cracks with wider gaps and bumps with higher elevation, which create a high level of vibrations, were misclassified as potholes.

Table 9. Summary of classification results using time-domain features and sparse features.

Type of damage	Number of samples	Precision rates (%)	
		Time-domain features	Sparse features
Pothole	155	76.12	80.65
Crack	34	44.11	52.94
Bump	20	30.00	45.00

5.4. Cloud-based Smart Phone App Development for Test and Validation

Finally, to encourage the adoption of the proposed method/system, and to validate the system in a larger area, we developed a Cloud-based smart phone app. The App, developed with JavaScript, does not require any installation for use. Figure 24 illustrates the screenshots of the app.

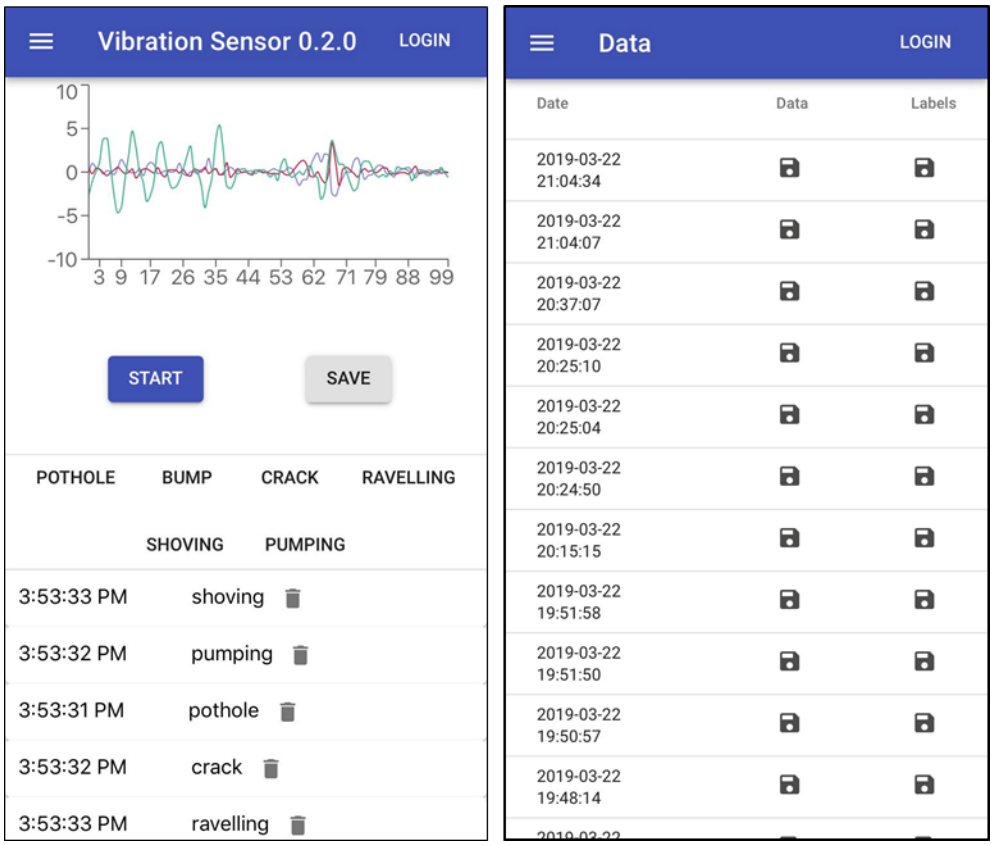

Figure 26. The cloud-based smart phone app for road damage detection using vehicle vibrations.

The App was developed using React, Node.js, and MongoDB.

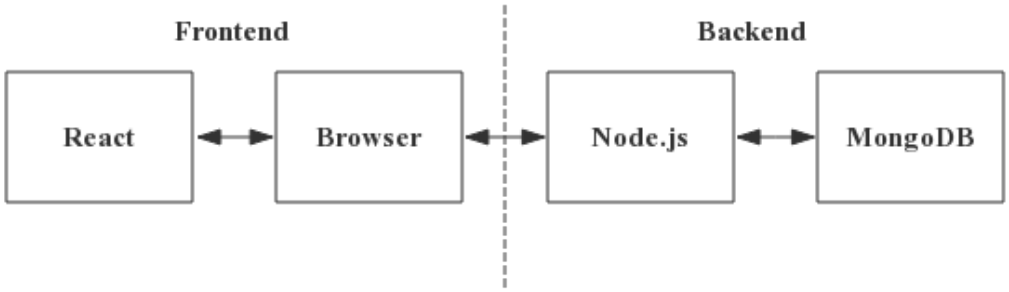

Figure 27. The architecture of the app.

6. CONCLUSIONS

This study proposed and tested an automated road pavement damage recognition method based on the vibration patterns of moving vehicles. The vibration features of regular moving vehicles were extracted to classify specific road pavement damages. To reduce the deployment cost and difficulties, regular phones equipped with accelerometers were used in vibration data collection. Compared to the special equipment used in similar vibration studies, using regular phones makes data collection much easier and broader, but it led to low-quality issues of data. 310 miles of road surface imagery data and corresponding vehicle vibration data in College Station, TX, and Baton Rouge, LA, were collected using a video camera and the smartphone application developed by the research team. The initial effort to use classic machine learning methods, including Support Vector Machine, Decision Tree, Neural Networks, provided the accuracies of 90.15%, 88.35%, and 91.90%. These methods reliably distinguished damaged surfaces from smooth surfaces, but they failed to provide a reliable classification between damaged surface types (i.e., pothole, crack, and bump) due to noises in vibration data.

To improve the effectiveness of using low-quality data, this study used Sparse Coding, a deep learning method that finds a sparse representation of the input raw data in the form of a linear combination of basic elements called bases or atoms. The dimensionality of the raw temporal vibration data is, therefore significantly reduced. The study was performed with vehicle vibration data collected in College Station, Texas. The study found that the classification model using the features extracted from Sparse Coding were able to classify potholes, cracks, and bumps at the accuracies of 80.65%. 52.94%, and 45%, respectively. The low accuracies in classifying cracks and bumps might be largely due to the imbalanced data samples and fewer instances on those damage types within the current dataset, and the planned data collection using crowdsourcing platforms would contribute to addressing this issue and enhancing the classification performance.

REFERENCES

1. Duggan, W. The True Cost Of Fixing America's Infrastructure.In *Market Watch*, 2017.

2. ASCE. *The ASCE Grand Challenge*. https://collaborate.asce.org/ascegrandchallenge/home.

3. America's Infrastructure Report Card 2017.In, 2017.

4. FHA. 2010 Status of the Nation's Highways, Bridges, and Transit: Conditions & Performance.In, 2010.

5. USDOT. National Motor Vehicle Crash Causation Survey.In, 2008.

6. Wei, L., S. Kayser, and F. Wellner. Impact of surface temperature on fatigue damage in asphalt pavement. *Journal of Highway and Transportation Research and Development (English Edition)*, 2013. 7, 3: 1-6.

7. Dekker, R. Applications of maintenance optimization models: a review and analysis. *Reliability Engineering & System Safety*, 1996. 51, 3: 229-240.

8. Kobayashi, K., M. Do, and D. Han. Estimation of Markovian transition probabilities for pavement deterioration forecasting. *KSCE Journal of Civil Engineering*, 2010. 14, 3: 343-351.

9. Peshkin, D., K. Zimmerman, T. Freeman, and K. Smith. Pavement Preservation: Preventive Maintenance Treatment, Timing, and Selection. *Instructor's Guide. Federal Highway Administration, Washington, DC*, 2007.

10. Chen, F., H. Yu, R. Hu, and X. Zeng. Deep learning shape priors for object segmentation.In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 1870-1877.

11. Medina, R., J. Gómez-García-Bermejo, and E. Zalama. Automated visual inspection of road surface cracks.In *International Symposium on Automation and Robotics in Construction (ISARC)*, 2010.

12. Pierce, L. M., G. McGovern, and K. A. Zimmerman. Practical guide for quality management of pavement condition data collection. 2013.

13. TXDOT. Pavement Management Information System - Rater's Manual.In, TX, 2016.

14. Cafiso, S., A. Di Graziano, O. Giudice, and G. Pappalardo. Tools for Road Inspection and Safety Management.In *3rd International Conference on Transportation Infrastructure, Pisa*, 2014.

15. Achenbach, J. On the road from schedule-based nondestructive inspection to structural health monitoring.In *6th International Workshop on Structural Health Monitoring: Quantification, Validation, and Implementation, IWSHM 2007*, DEStech Publications, 2007.

16. Varadharajan, S., S. Jose, K. Sharma, L. Wander, and C. Mertz. Vision for road inspection.In *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, IEEE, 2014. 115-122.

17. Ferguson, R. A., D. N. Pratt, P. R. Turtle, I. B. Macintyre, D. P. Moore, P. D. Kearney, M. J. Best, J. L. Gardner, M. Berman, and M. J. Buckley. Road pavement deterioration inspection system.In, Google Patents, 2003.

18. Salari, E., and G. Bao. Automated pavement distress inspection based on 2D and 3D information.In *Electro/Information Technology (EIT), 2011 IEEE International Conference on*, IEEE, 2011. 1-4.

19. Huang, Y., and B. Xu. Automatic inspection of pavement cracking distress. *Journal of Electronic Imaging*, 2006. 15, 1: 013017-013017-013016.

20. Pu, S., M. Rutzinger, G. Vosselman, and S. O. Elberink. Recognizing basic structures from mobile laser scanning data for road inventory studies. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2011. 66, 6: S28-S39.

21. Mahmoudzadeh, M., J.-B. Got, S. Lambot, and C. Grégoire. Road inspection using full-wave inversion of far-field ground-penetrating radar data.In *Advanced Ground Penetrating Radar (IWAGPR), 2013 7th International Workshop on*, IEEE, 2013. 1-6.

22. Su, Y.-S., S.-C. Kang, J.-R. Chang, and S.-H. Hsieh. Dual-light inspection method for automatic pavement surveys. *Journal of Computing in Civil Engineering*, 2012. 27, 5: 534-543.

23. Yu, S.-J., S. R. Sukumar, A. F. Koschan, D. L. Page, and M. A. Abidi. 3D reconstruction of road surfaces using an integrated multi-sensory approach. *Optics and lasers in engineering*, 2007. 45, 7: 808-818.

24. Zhang, L., F. Yang, Y. D. Zhang, and Y. J. Zhu. Road crack detection using deep convolutional neural network.In *Image Processing (ICIP), 2016 IEEE International Conference on*, IEEE, 2016. 3708-3712.

25. Ouyang, W., and B. Xu. Pavement cracking measurements using 3D laser-scan images. *Measurement Science and Technology*, 2013. 24, 10: 105204.

26. Laurent, J., J. F. Hébert, D. Lefebvre, and Y. Savard. Using 3D laser profiling sensors for the automated measurement of road surface conditions.In *7th RILEM International Conference on Cracking in Pavements*, Springer, 2012. 157-167.

27. Wilde, W. J. Implementation of an international roughness index for Mn/DOT pavement construction and rehabilitation.In, 2007.

28. Smith, K., K. Smith, L. Evans, T. Hoerner, M. Darter, and J. Woodstrom. Smoothness specifications for pavements.In, 1997.

29. Finn, F. Pavement management systems--Past, present, and future. *Public Roads*, 1998. 62, 1.

30. Sayers, M. W. The international road roughness experiment: Establishing correlation and a calibration standard for measurements. 1986.

31. ASTM, E. 98, Standard Practice for Computing International Roughness Index OF Roads from Longitudinal Profile Measurements. *American Society for Testing and Materials, Philadelphia, PA*, 2003.

32. Gillespie, T. D. Everything you always wanted to know about the iri, but were afraid to ask. *The University of Michigan Transportation Research Institute. Nebraska,* 1992.

33. Hanson, T., C. Cameron, and E. Hildebrand. Evaluation of low-cost consumer-level mobile phone technology for measuring international roughness index (IRI) values. *Canadian Journal of Civil Engineering,* 2014. 41, 9: 819-827.

34. Seraj, F., B. J. van der Zwaag, A. Dilo, T. Luarasi, and P. Havinga. RoADS: A road pavement monitoring system for anomaly detection using smart phones.In *Big data analytics in the social and ubiquitous context,* Springer, 2014. 128-146.

35. Douangphachanh, V., and H. Oneyama. A study on the use of smartphones for road roughness condition estimation. *Journal of the Eastern Asia Society for Transportation Studies,* 2013. 101551-1564.

36. Douangphachanh, V., and Oneyama, H. Estimation of road roughness condition from smartphones under realistic settings.In *ITS Telecommunications (ITST), 2013 13th International Conference on,* IEEE, 2013. 433-439.

37. Sayers, M., and S. J. A. A. Karamihas, MI. The Little Book of Profiling, University of Michigan Transportation Research Institute. 1998.

38. Forslöf, L., and H. Jones. Roadroid: Continuous road condition monitoring with smart phones. *Journal of Civil Engineering Architecture,* 2015. 9, 4: 485-496.

39. Li, X., and D. W. Goldberg. Toward a mobile crowdsensing system for road surface assessment. *Computers, Environment Urban Systems,* 2018. 6951-62.

40. Lepine, J., V. Rouillard, and M. Sek. On the use of machine learning to detect shocks in road vehicle vibration signals. *Packaging Technology Science,* 2017. 30, 8: 387-398.

41. Allouch, A., A. Koubâa, T. Abbes, and A. Ammar. Roadsense: Smartphone application to estimate road conditions using accelerometer and gyroscope. *IEEE Sensors Journal,* 2017. 17, 13: 4231-4238.

42. Bhoraskar, R., N. Vankadhara, B. Raman, and P. Kulkarni. Wolverine: Traffic and road condition estimation using smartphone sensors.In *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012),* IEEE, 2012. 1-6.

43. Silva, N., J. Soares, V. Shah, M. Y. Santos, and H. Rodrigues. Anomaly detection in roads with a data mining approach. *Procedia computer science,* 2017. 121415-422.

44. Douangphachanh, V., and H. Oneyama. A Model For The Estimation Of Road Roughness Condition From Sensor Data Collected By Android Smartphones. *Proceedings of Civil Engineering Society,* 2014. 70, 5: I_103-I_111.

45. Singh, G., D. Bansal, S. Sofat, and N. Aggarwal. Smart patrolling: An efficient road surface monitoring using smartphone sensors and crowdsourcing. *Pervasive Mobile Computing,* 2017. 4071-88.

46. Seraj, F., B. J. van der Zwaag, A. Dilo, T. Luarasi, and P. Havinga. RoADS: A road pavement monitoring system for anomaly detection using smart phones.In *Big data analytics in the social and ubiquitous context*, Springer, 2015. 128-146.

47. Chen, K., M. Lu, X. Fan, M. Wei, and J. Wu. Road condition monitoring using on-board three-axis accelerometer and GPS sensor.In *2011 6th International ICST Conference on Communications and Networking in China (CHINACOM)*, IEEE, 2011. 1032-1037.

48. Mohan, A., and S. Poobal. Crack detection using image processing: A critical review and analysis. *Alexandria Engineering Journal*, 2018. 57, 2: 787-798.

49. Tedeschi, A., and F. Benedetto. A real-time automatic pavement crack and pothole recognition system for mobile Android-based devices. *Advanced Engineering Informatics*, 2017. 3211-25.

50. Masino, J., Thumm, J., Frey, M., & Gauterin, F. Learning from the crowd: Road infrastructure monitoring system. *Journal of Traffic and Transportation Engineering (English Edition), 4(5), 451-463.*, 2017.

51. Howe, J. The rise of crowdsourcing. *Wired magazine*, 2006. 14, 6: 1-4.

52. Nickerson, J. V., and Y. Sakamoto. Crowdsourcing creativity: Combining ideas in networks.In *Workshop on information in networks*, 2010.

53. Staffelbach, M., P. Sempolinski, D. Hachen, A. Kareem, T. Kijewski-Correa, D. Thain, D. Wei, and G. Madey. Lessons learned from an experiment in crowdsourcing complex citizen engineering tasks with Amazon Mechanical Turk. *arXiv preprint arXiv:1406.7588*, 2014.

54. Gerth, R. J., A. Burnap, and P. Papalambros. Crowdsourcing: A primer and its implications for systems engineering.In, DTIC Document, 2012.

55. Schenk, E., and C. Guittard. Crowdsourcing: What can be Outsourced to the Crowd, and Why.In *Workshop on Open Source Innovation, Strasbourg, France*, 2009. 72.

56. Zhai, Z., P. Sempolinski, D. Thain, G. Madey, D. Wei, and A. Kareem. Expert-Citizen Engineering:" Crowdsourcing" Skilled Citizens.In *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, IEEE, 2011. 879-886.

57. Van Dyk, D. A., and X.-L. Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 2001. 10, 1: 1-50.

58. Pan, S. J., and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2010. 22, 10: 1345-1359.

59. Raina, R., A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data.In *Proceedings of the 24th international conference on Machine learning*, ACM, 2007. 759-766.

60. Lee, H., R. Raina, A. Teichman, and A. Ng. Exponential family sparse coding with applications to self-taught learning.In *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

61. Gan, J., L. Li, Y. Zhai, and Y. Liu. Deep self-taught learning for facial beauty prediction. *Neurocomputing*, 2014. 144295-303.

62. Jie, Z., Y. Wei, X. Jin, J. Feng, and W. Liu. Deep self-taught learning for weakly supervised object localization.In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 1377-1385.